Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

1961

Engineering design of logical circuits.

Farrell, John R.

Monterey, California: U.S. Naval Postgraduate School

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

NPS ARCHIVE
1961
FARRELL, J.

ENGINEERING DESIGN OF LOGICAL CIRCUITS

JOHN R. FARRELL

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

Mat 117

ENGINEERING DESIGN OF LOGICAL CIRCUITS

* * * * * * *

John R. Farrell

[illegible] Postgraduate School
[illegible] California

24 August 1961

. G. N. Grashorn
ata Display, Incorporated
320 Como Avenue
t. Paul 4, Minnesota

ear Mr. Grashorn:

LT John R. Farrell, USN, who performed his industrial training tour ith your firm has indicated that he has presented a copy of his thesis ntitled "Engineering Design of Logical Circuits" to you.

Our normal policy is to distribute copies of theses to any office of e Department of Defense who desires a copy and to permit civilian industrial rms and other offices of the government to borrow copies on request. Before king this thesis available on loan request to other than Department of efense establishments, we would appreciate an indication from you that your mpany has no objection to such loan.

This request for release is made in order to assure us that such loans o other than Department of Defense activities will not disclose any proietary information of your company.

I am enclosing a copy of this letter with a release certificate at the ottom which may be signed and returned to us in the enclosed envelope.

Very sincerely yours,

ENGINEERING DESIGN OF LOGICAL CIRCUITS

by

John R. Farrell
//
Lieutenant, United States Navy

Submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE
IN
ENGINEERING ELECTRONICS

United States Naval Postgraduate School
Monterey, California

1 9 6 1

Library
U. S. Naval Postgraduate School
Monterey, California

ENGINEERING DESIGN OF LOGICAL CIRCUITS

by

John R. Farrell

This work is accepted as fulfilling

the thesis requirements for the degree of

MASTER OF SCIENCE

in

ENGINEERING ELECTRONICS

from the

United States Naval Postgraduate School

ABSTRACT

Over the past decade many valuable "tools" have been developed for the logical designer. Most of this development has been quite formal. From a practical point of view, it is often difficult to apply the "tools" to the actual design of equipment. However if these "tools" are developed in terms of the basic circuit module of a particular piece of equipment they may be used to greatly simplify the design of the equipment. For example, the symbolic logic diagram can be used to produce a finished circuit design if symbols are chosen which represent the basic circuit module, and circuit as well as logical constraints are considered in drawing the diagram. Even in the case of Boolean algebra a form of expression can be chosen, based on the circuit module, which simplifies the design of the final logical circuitry.

The author wishes to express his appreciation to Professor Mitchell L. Cotton of the U.S. Naval Postgraduate School, and to Mr. G. N. Grashorn of Data Display, Inc. for their many helpful suggestions.

TABLE OF CONTENTS

Section	Title	Page
1.	Introduction	1
2.	Basic Logic Circuit	3
2.1	Basic Logic Circuit	3
2.1.1	OR Circuit	3
2.1.2	Inverter Circuit	4
2.1.3	The AND Connection	7
2.2	Logic Cards	7
2.3	Rules Governing the Use of the Logic Circuit	10
3.	Logic Diagram	19
3.1	Basic Inverter Symbol	19
3.2	Flip-flop	22
3.3	Solution of Design Problems Using the Logic Diagram	25
3.3.1	The Design of the Decoding Circuitry	25
3.3.2	The Execution of an Instruction Sequence	31
4.	Derivation of an Optimum Boolean Expression	39
4.1	The Simple Binary Adder	39
5.	Transcription of the Logical Design Into Wiring Instructions	44
5.1	File of Equations	44
5.2	Chassis Map	47
5.3	Wiring Tabulation	47
6.	Example Design Problem	53
6.1	Logic Diagram and Circuit Description	53
6.2	File of Equations	61

TABLE OF CONTENTS
(Continued)

Section	Title	Page
6.3 Chassis Map		63
6.4 Wire Tabulation		64
7. Conclusion		66
Bibliography		67
Appendix A		68

LIST OF ILLUSTRATIONS

Figure		Page
2-1	CDC Basic Logic Circuit	5
2-2	CDC Basic Logic Circuit	6
2-3	Single Output to Single Input Connection	8
2-4	Input = A·B·C	8
2-5	Single Inverter Card 14	9
2-6	Double Inverter Card 22	9
2-7	Single Inverter Card 14	12
2-8	Single Inverter Card 16	13
2-9	Double Inverter Card 21	14
2-10	Double Inverter Card 22	15
2-11	Double Inverter Card 23	16
2-12	Double Inverter Card 24	17
2-13	Table 2-1, Description of Standard Card Types	18
3-1	Typical Logic Symbols	20
3-2	Basic Inverter	21
3-3	Basic Inverter Symbol	21
3-4	Logic Diagram	23
3-5	Logic Diagram	23
3-6	Interconnection of Inverters to Form a Flip-flop	24
3-7	Logic Diagram Symbol for A Flip-flop	24
3-8	1604 Instruction Decoding Circuit	27
3-9	Table 3-1, Program Instructions	28
3-10	160 Instruction Decoding Circuit	30
3-11	160 Request Status Response	33

LIST OF ILLUSTRATIONS
(Continued)

Figure		Page
3-12	1604 Interrupt Response	34
3-13	1604 Sense Response	36
3-14	Gating 1604 Data into 160	37
4-1	Binary Addition with Seven Inverters	40
4-2	Binary Addition with Six Inverters	40
4-3	Binary Addition with Four Inverters	42
5-1	Format of Equation File	45
5-2	Example Logic Diagram	45
5-3	Chassis Map	48
5-4	File of Equation Entries with Wire Tab Numbers	50
5-5	Typical Wire Tabulation	50
6-1	Block Diagram of Binary Adder	54
6-2	Logic Diagram of Pulse and Counter Circuits	55
6-3	Observed Pulse at Output of J^{008}	57
6-4	HP 211A Output and Pulses from J^{008} and J^{010}	58
6-5	Pulse from J^{008} and Counter Output	59
6-6	Adder Logic Diagram	60

1. Introduction

During the past few years the more formal aspects of the logical design of digital computer circuits have received fairly widespread attention [1-5]. In spite of this, the current state of the art is such that the actual engineering logical design of even the most recent large-scale digital computers still relies most heavily on methods that are only rather tenuously related to the formal theories. The present study has been undertaken with the objective of narrowing the gap between the engineering design and classical logical design in the environment of a particular circuit philosophy.

The methods developed in this paper were the outgrowth of a design project involving logical circuitry for the digital "interface" between a large-scale digital computer, a small computer, and a radar data transfer and operator's control console. Such a unit is currently under development for the U.S. Naval Postgraduate School's Digital Control and Automation Laboratory. This "interface" design is described in its entirety in a separate engineering report. [10]

The design procedure for the construction of logical equipment can be divided into four parts; (1) the development of a basic logic circuit; (2) the selection of a logic diagram symbol to best represent the circuitry; (3) the derivation of an optimum Boolean expression to represent the logical circuitry; and (4) the development of a procedure for transcribing the Boolean expressions into wiring instructions.

The development of a basic logic circuit is based on the consideration of such factors as switching speed, circuit timing, cost, etc. The basic logic circuit used in the design of the logical "interface"

mentioned above was developed by the Control Data Corporation (CDC) and is the basic circuit used in their computers. This circuit will be described and used as the basis for the development of the design procedure. This same procedure can be applied to any basic logic circuit chosen by the designer.

The principles developed are illustrated by examples taken from the logical "interface" design, and from a smaller problem which is amenable to construction in the laboratory. Appendix A contains a reference list of fundamental Boolean relations.

2. Basic Logic Circuit

The CDC logic circuit will be used to illustrate the design procedure presented in this paper. The circuit is described in section 2.1. The basic logic circuit is mounted in several configurations on a 15 pin logic card. The logic card is described in section 2.2. A set of rules which govern the use of the logic card are delineated in section 2.3.

2.1 Basic Logic Circuit

The basic logic circuit is shown in figure 2-1. This circuit requires three supply voltages; -20, 0, +20 volts. A -3.0 volts at an input or output terminal represents a logical "1"; a -0.5 volts a logical "0". The circuit will switch between these levels in not less than 50 nanoseconds and not more than 100 nanoseconds. The logic of the circuitry is OR - NOT - AND.

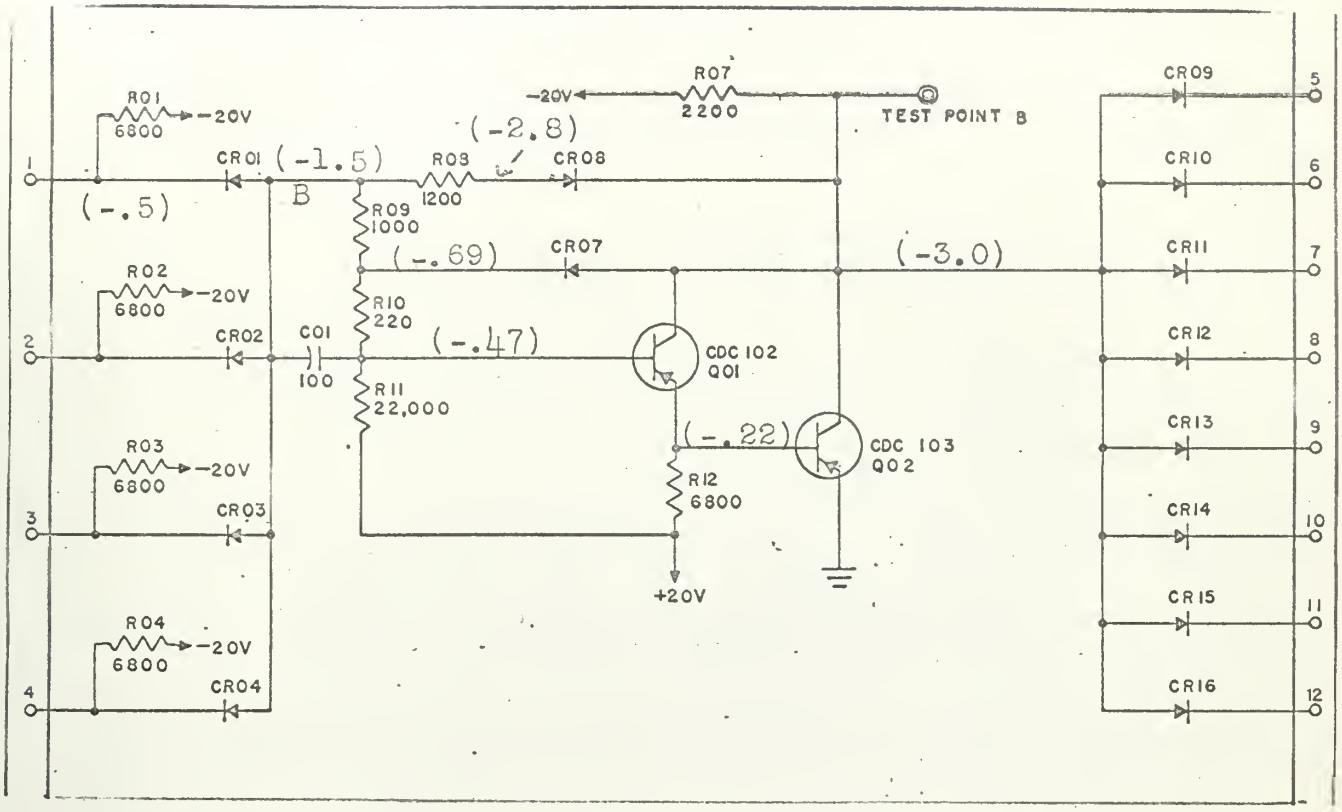
2.1.1 OR Circuit

In figure 2-1 R01, CR01, and R02, CR02 form two separate OR inputs. As many as six OR input circuits may be used with a single inverter circuit. If a logical "0" (-0.5v) is applied at each of the input terminals of the OR circuits the voltage at point B will be -1.5 volts, none of the diodes CR01, CR02, etc., will conduct, and the inverter circuit will interpret the input as a logical "0". If any of the input terminals go to a logical "1" (-3.0v), the diodes at these input terminals will conduct and the voltage at point B will drop to approximately -2.7 volts. The inverter circuit will recognize this voltage as an input logical "1".

2.1.2 Inverter Circuit

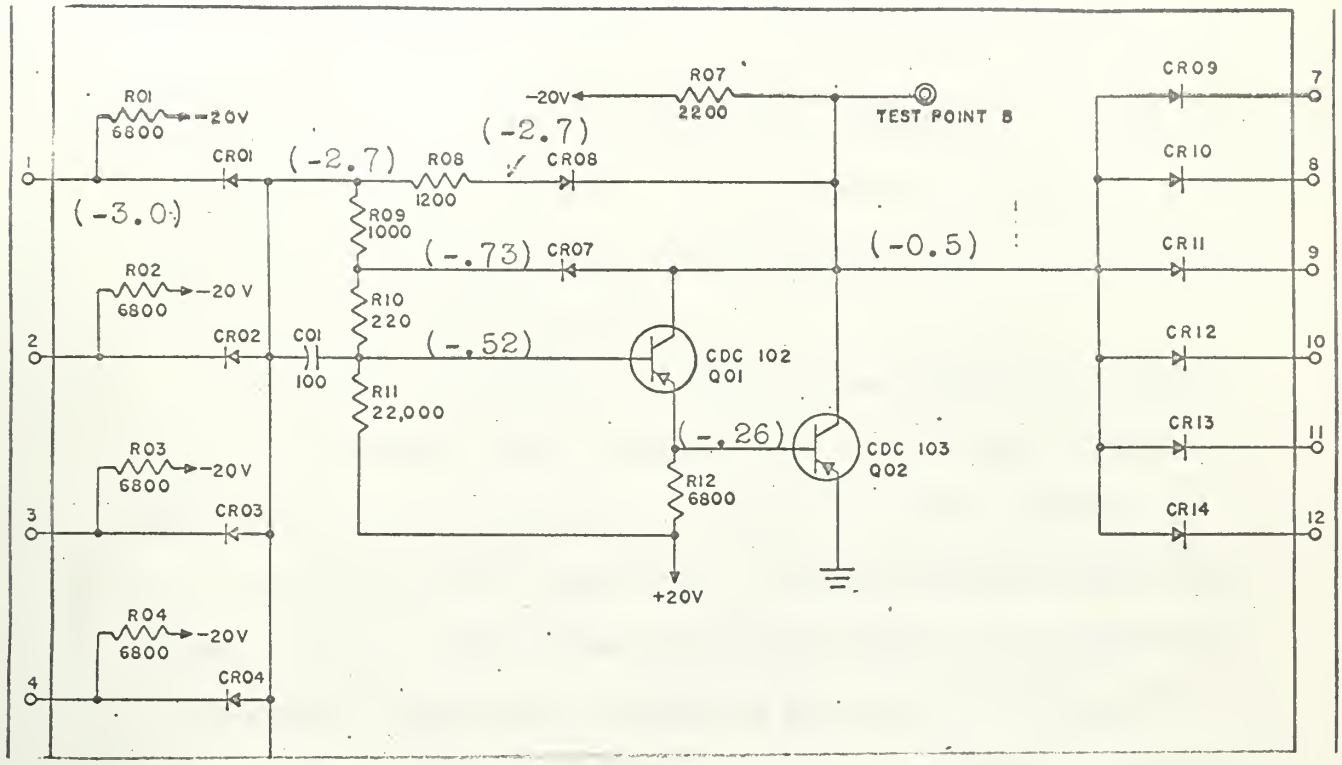
This part of the logic circuitry performs the NOT operation. In the circuit an emitter follower (Q01) drives a grounded emitter amplifier. A capacitor (C01) is used at the input to the inverter circuit to improve the switching time. Consider the circuit in figure 2-1 when the inputs to all the OR circuits are a logical "0". The voltages at the junction points in the circuit are indicated in the parentheses; all voltages are measured with respect to ground. CR01 and CR07 are back-biased, and therefore represent open circuits. CR08 is a feedback diode and is conducting. This diode sets up a voltage divider network, R08, R09, R10, and R11, between the output voltage and the +20 volt supply. This sets the voltage at the base of Q01 at -0.47 volts, and at the base fo Q02 at -0.22 volts. The voltage at the collector is -3.0 volts, or a logical "1"; the desired output.

Referring now to figure 2-2, consider the inverter circuit when the input to any one of the OR circuits is a logical "1" (-3.0v). CR01 and CR07 are conducting, and CR08 is back-biased. Again the voltages at the various junction points in the circuit are indicated in parentheses. A voltage divider network consisting of R01, R09, R10, and R11 is set up between the supply voltages, -20 volts and +20 volts. The voltage division in this network is affected by the feedback through CR07. The voltage at the base of Q01 goes more negative causing increased conduction through this transistor. This causes more voltage drop across R12, and the voltage at the base of Q02 goes more negative to a -0.26 volts. The voltage at the collectors of the two transistors rises to -0.5 volts, a logical "0".



NOTE: Circuit voltage levels for an input logical "0" are indicated in parentheses.

Figure 2-1 CDC Basic Logic Circuit



NOTE: Circuit voltage levels for an input logical "1" are indicated in parentheses.

Figure 2-2 CDC Basic Logic Circuit

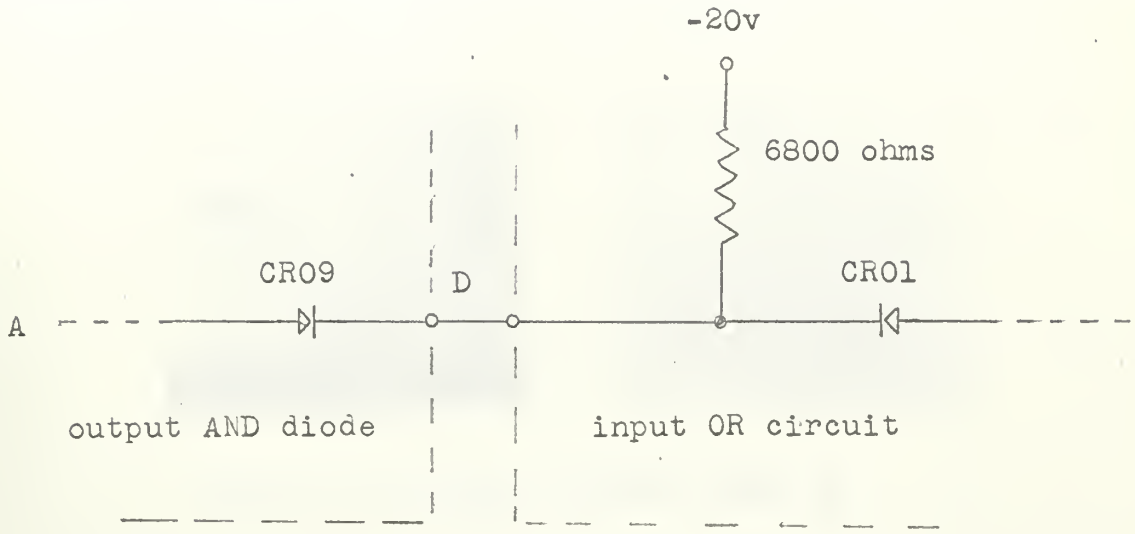
2.1.3 The AND Connection

The inverter output at the collectors is connected to the output terminals of the card through output diodes. Up to six output diodes are used on some cards. These diodes form AND circuits when considered in conjunction with the 6800 ohm resistor of the OR circuits at the input to the following inverter.

Consider the case where an output from one inverter is to be the only input to another inverter circuit. This is illustrated in figure 2-3. Diode CR01 may be considered an open circuit. Either a logical "1" or a logical "0" at A will cause CR09 to conduct and this voltage will become the input voltage to the next inverter circuit. Thus CR09 has no appreciable affect on the operation of the circuit. Now consider the case shown in figure 2-4 where the input to an inverter is to depend on the result of ANDing the outputs of three other inverter circuits. If the output of each of the inverters, A,B, and C, is a logical "1" (-3.0v) then all of the AND diodes will conduct and the OR circuit will see an input "1". But if any one of the inverters, A,B, or C, has an output which is "0" (0.5v), then the AND diode for this inverter alone will conduct. The voltage at point D will be -0.5 volts. The AND diodes for the other two inverter circuits are clearly back-biased. Thus the OR circuit will see an input of "0", and indeed the input to the OR circuit will depend on the Boolean expression $A \cdot B \cdot C$.

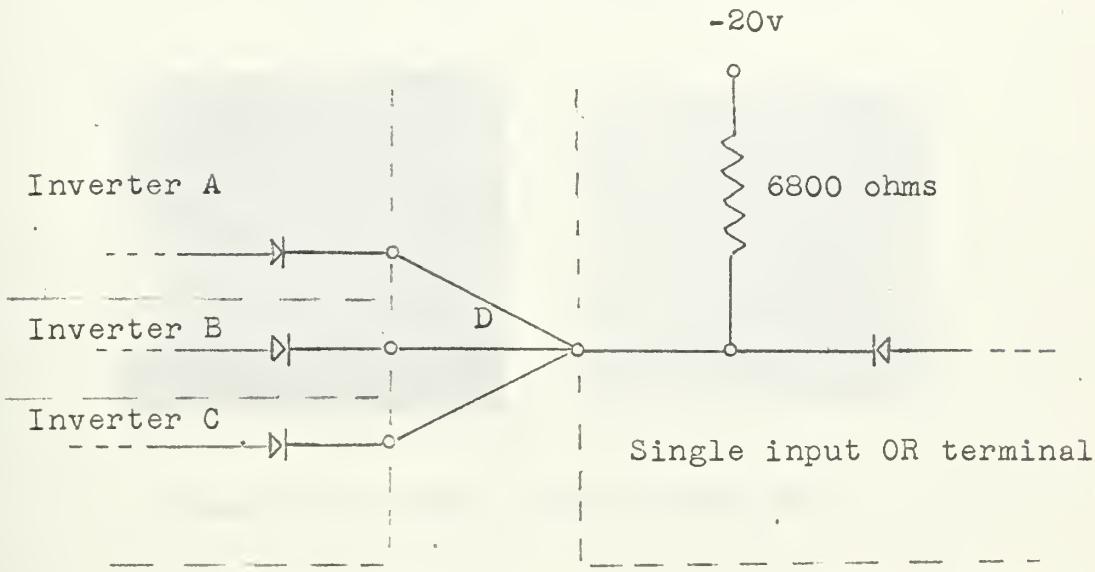
2.2 Logic Cards

The logic circuitry which was just discussed in the preceding section is mounted on a 15 pin printed circuit card. Two examples of this card are shown in figure 2-5 and figure 2-6. When looking at the



Single Output to Single Input Connection

Figure 2-3



Input = A·B·C

Figure 2-4

Figure 2-5 Single Inverter Card 14

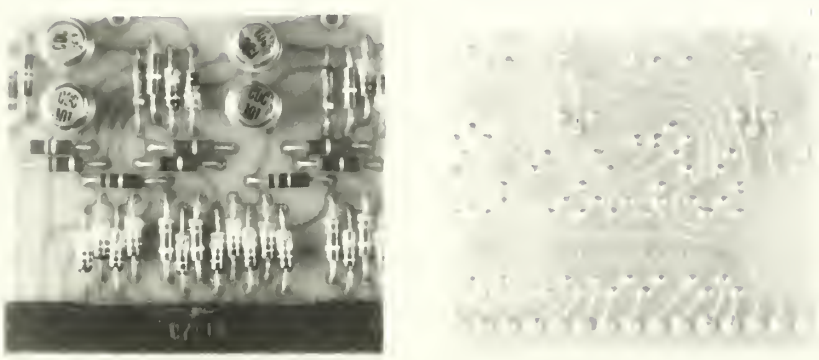


Figure 2-6 Double Inverter Card 22

component side of the card the pins are numbered from right to left, 1 to 15. Pins 13, 14, and 15 always carry the supply voltages; pin 13 is connected to -20 volts, pin 14 to 0 volts (ground), and pin 15 to +20 volts. Each type of CDC logic card carries an identifying number which is imprinted on the printed circuit side of the card. The types of logic cards which will be considered in this paper are numbers 14, 16 21, 22, 23, and 24. The circuit diagram and layout for each card are shown in figures 2-7 through 2-12. The highest order digit in each card number indicates the number of inverter circuits mounted on the card. That is the 14 and 16 cards each have one inverter circuit per card; the 21, 22, 23, and 24 cards each have two inverter circuits. The lower order digit indicates the number of OR inputs associated with each inverter circuit on the card. On the 15 pin card, since three pins (13, 14, and 15) carry supply voltages, this leaves 12 pins to be divided between input and output. Thus a 14 card has one inverter circuit, four input connections, and eight output connections. The lower numbered pins are always assigned to input. The 22 card has two inverter circuits, two inputs per inverter circuit, and four output connections for each inverter circuit. The pin assignments for each card type are summarized in table 2-1, page 18.

A test point for each inverter is located at the top of the card. The voltage at this test point is the transistors' collector voltage.

2.3 Rules Governing the Use of the Logic Circuit

The following rules apply to the use of the logic circuit.

(1) A maximum of eight outputs may be taken from a single inverter.

(2) A maximum of six simultaneously-gated outputs may be taken from a single inverter. Here gating implies the AND function.

(3) The number of OR inputs to an inverter is limited to a maximum of six.

(4) The maximum number of AND connections which can be made to a single OR input is four.

(5) The switching time of a single inverter is from 50 to 100 nanoseconds. An allowance must be made for this switching time for each inversion.

(6) The total number of inputs and outputs of a single inverter must not exceed 12.

(7) Any unused input terminals on a card must be grounded.

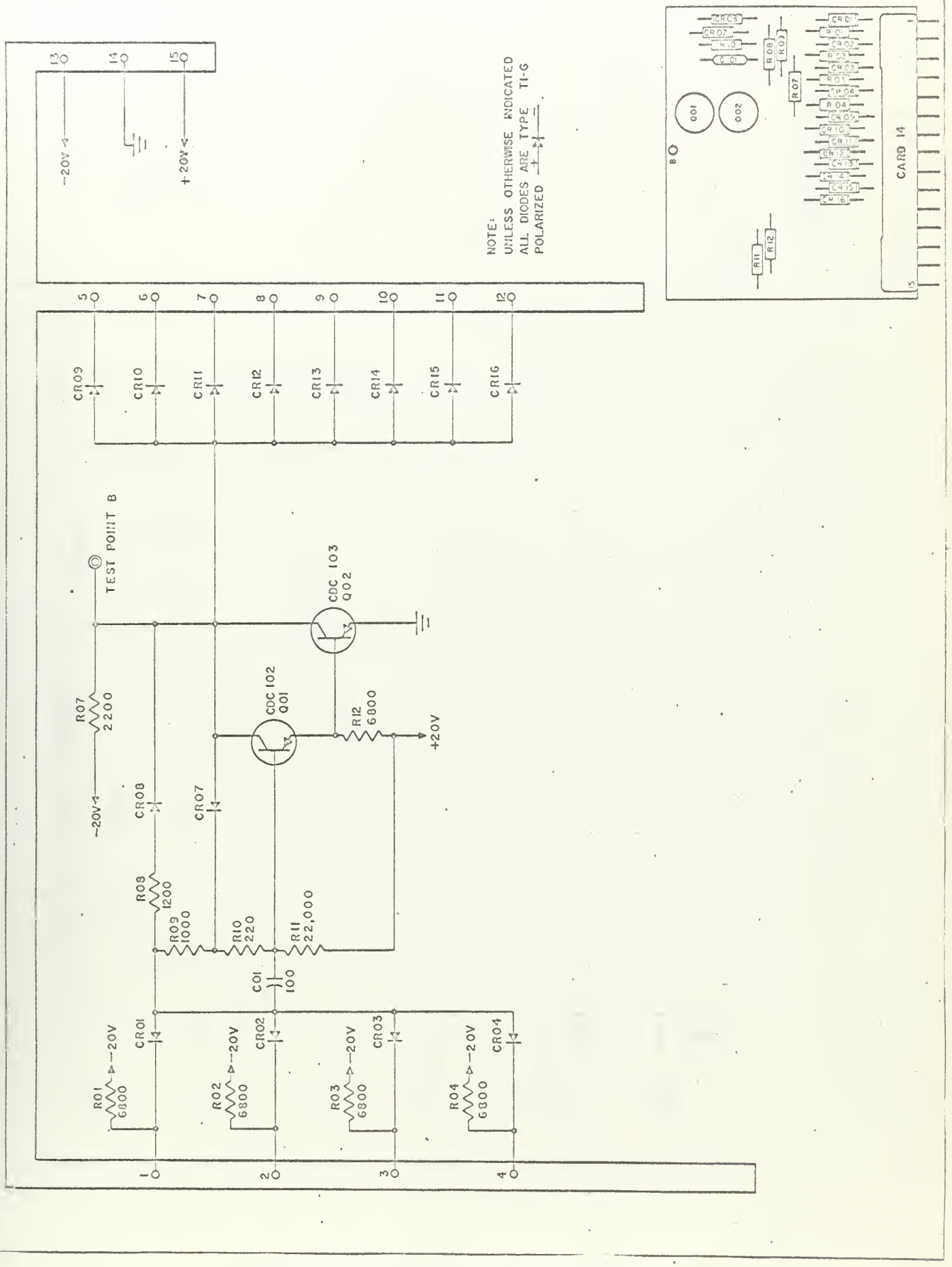


Figure 2-7 Single Inverter Card 14

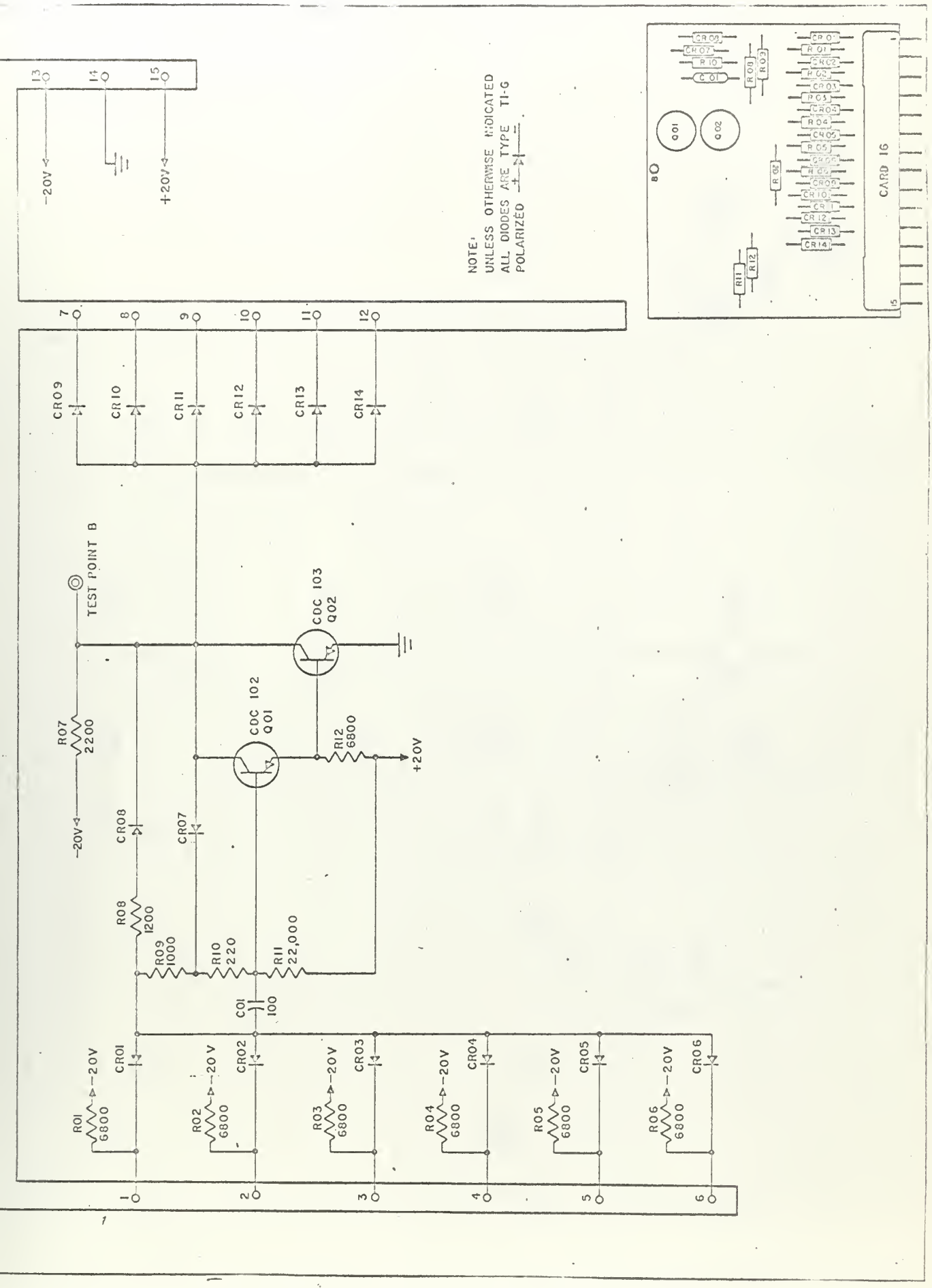


Figure 2-8 Single Inverter Card 16

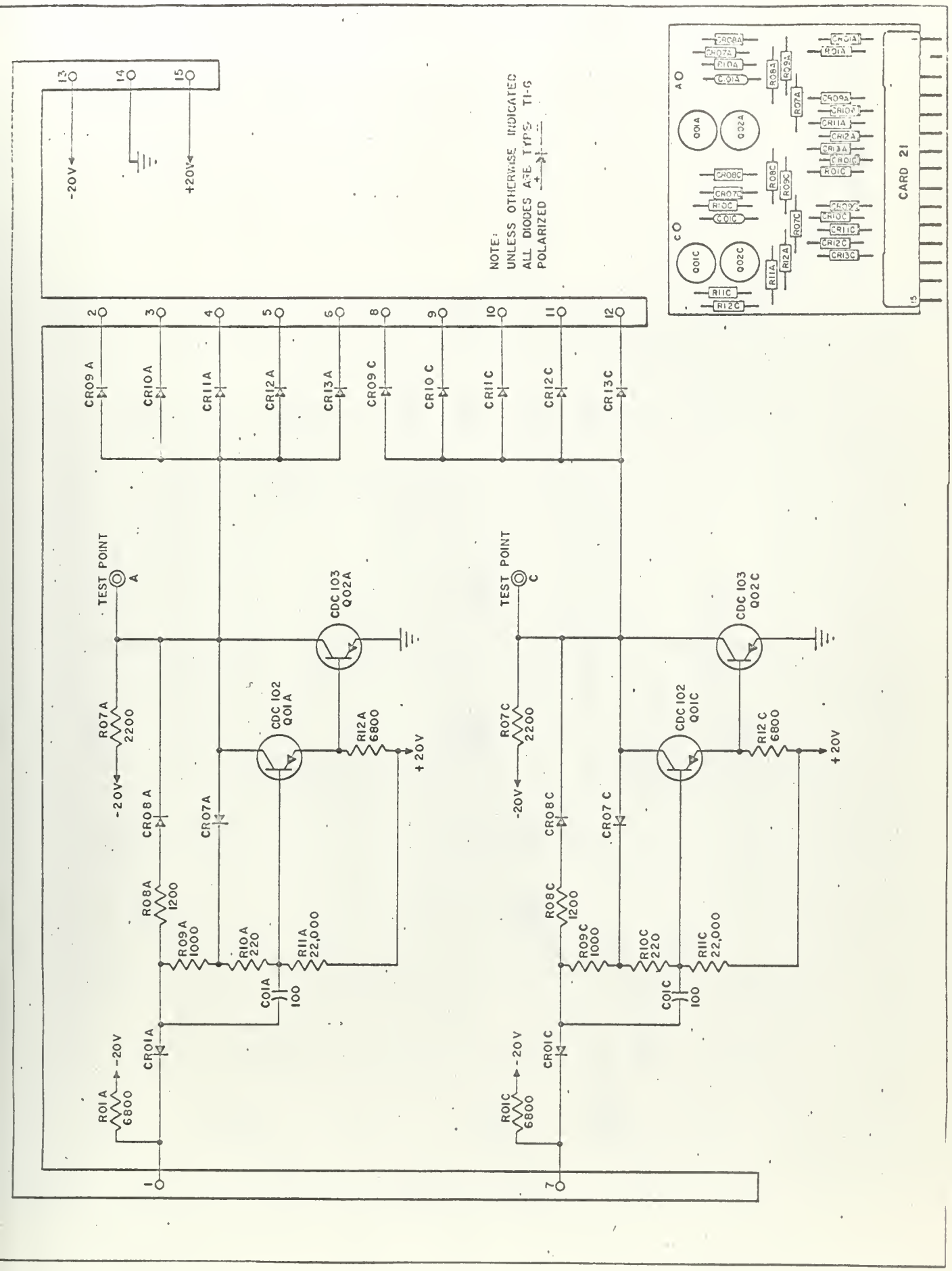


Figure 2-9 Double Inverter Card 21

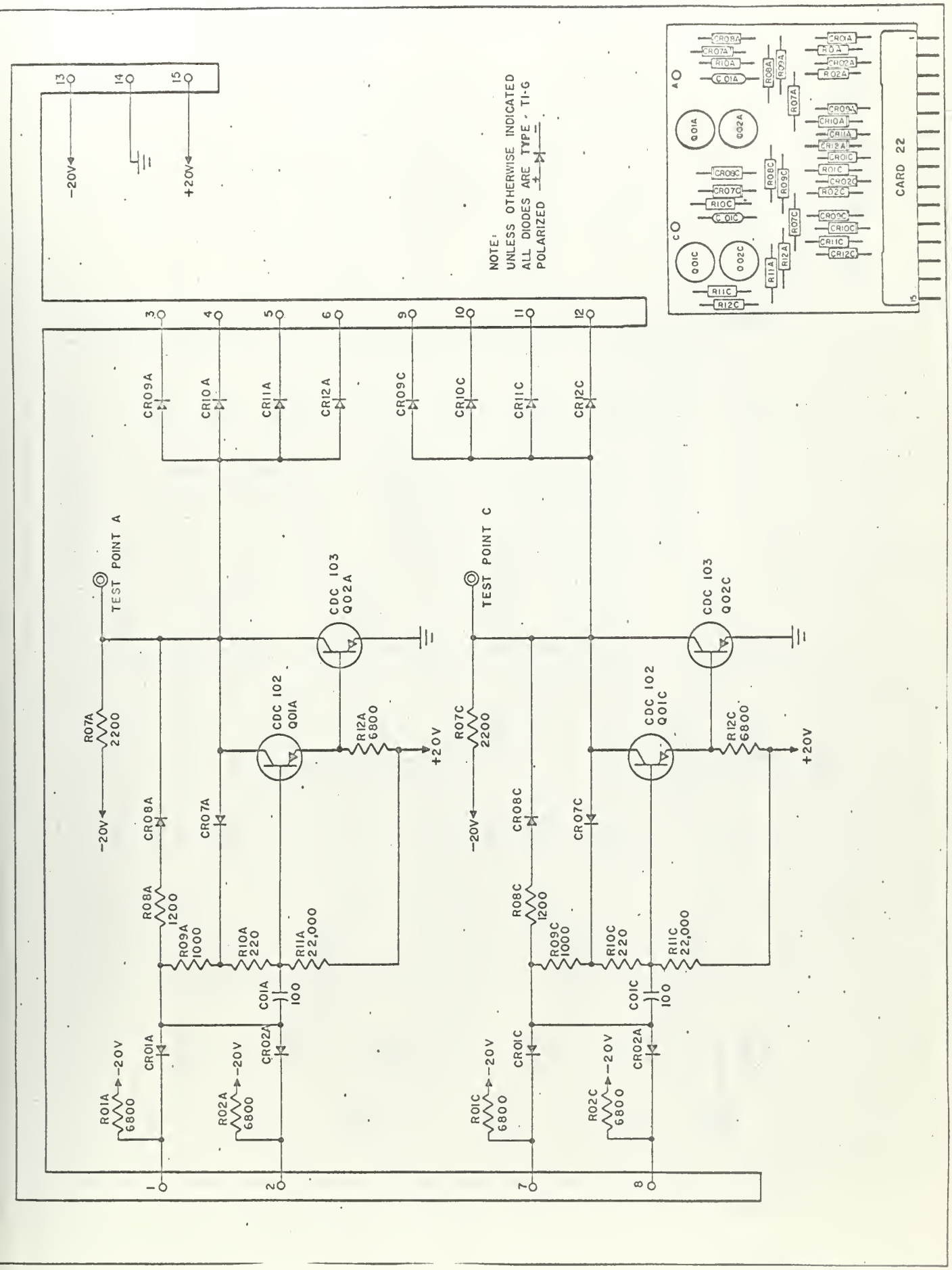


Figure 2-10 Double Inverter Card 22

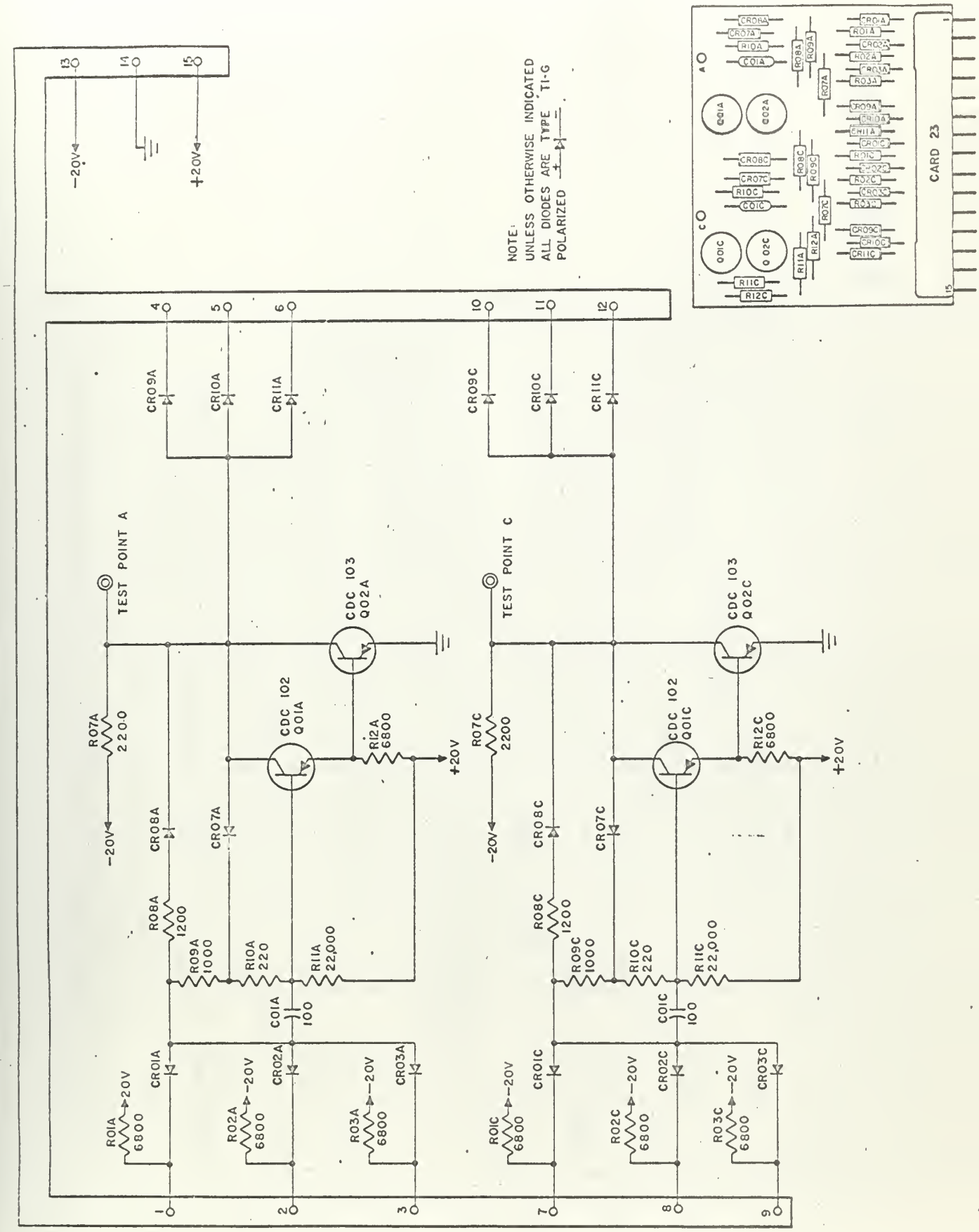


Figure 2-11 Double Inverter Card 23

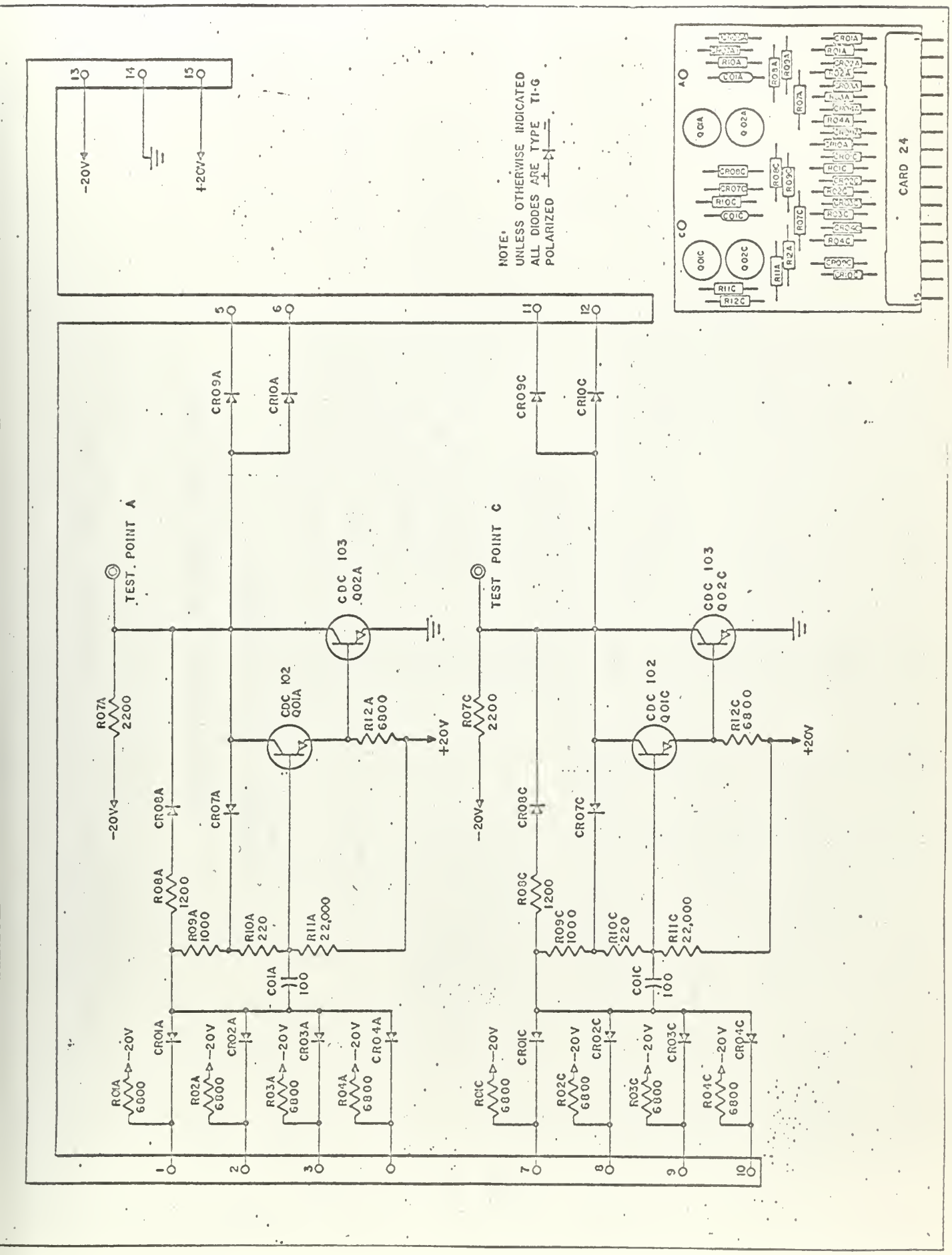


Figure 2-12 Double Inverter Card 24

Figure 2-13 TABLE 2-1. DESCRIPTION OF STANDARD CARD TYPES

Type Designation	No. of Inverters	No. of inputs (per inverter)	No. of outputs (per inverter)	1	2	3	4	5	6	7	8	9	10	11	12
14	1	4	8	I	I	I	I	O	O	O	O	O	O	O	O
16	1	6	6	I	I	I	I	I	I	O	O	O	O	O	O
21	2	1	5	I_A	O_A	O_A	O_A	O_A	O_A	I_C	O_C	O_C	O_C	O_C	O_C
22	2	2	4	I_A	I_A	O_A	O_A	O_A	O_A	I_C	I_C	O_C	O_C	O_C	O_C
23	2	3	3	I_A	I_A	I_A	O_A	O_A	O_A	I_C	I_C	I_C	O_C	O_C	O_C
24	2	4	2	I_A	I_A	I_A	I_A	O_A	O_A	I_C	I_C	I_C	I_C	O_C	O_C

Supply Voltages: (Same for all cards)

Pin	13	14	15
	-20v	Gd	+20v

3. Logic Diagram

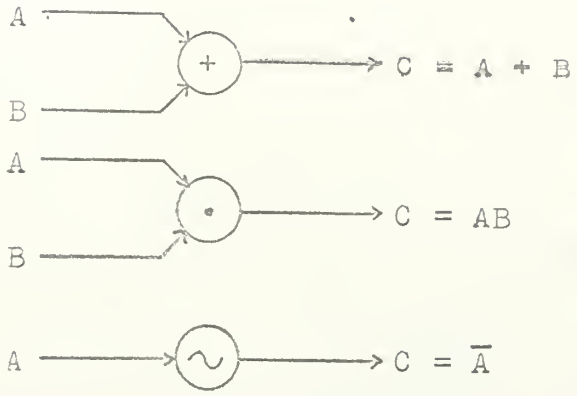
The logic diagram is a symbolic representation of a Boolean expression. The use of this symbolism is now standard practice in any treatise on logical design /6,7,8/. The symbols in use are quite non-standard and vary considerably from one author to another. Some typical symbols taken from the work of three authors are shown in figure 3-1. From an engineering design point of view such symbols are practically useless. If however a symbolic notation is selected which represents as nearly as possible the basic logic circuit then quite often reasonably complex logical design problems can be solved directly using this notation.

A symbolic notation to represent the basic inverter circuit is developed in section 3.1. In section 3.2 this notation is extended to include the flip-flop. The direct solution of a design problem is illustrated in section 3.3 by an example taken from the design of the digital "interface" mentioned in the Introduction.

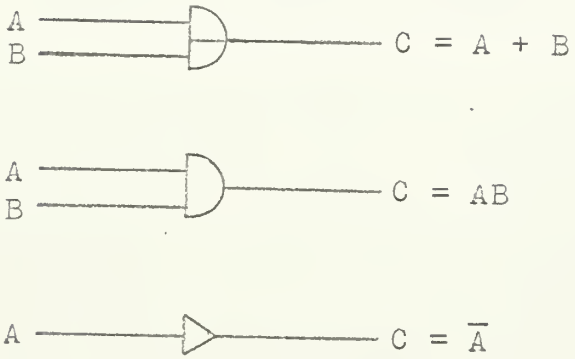
3.1 Basic Inverter Symbol

The unit of logical design is the basic inverter circuit described in section 2. This unit has an OR circuit at each input pin and AND diodes at each output pin. The basic inverter will be represented by a rectangle in the logic diagram; see figure 3-2. An arrow drawn into this rectangle will represent an input which is to have an OR relation with the other input arrows. This is represented in figure 3-3. The AND connection is made by hooking together the input terminal and all of the output terminals to be ANDed. This AND connection

Caldwell [6], p. 321.



Ledley [7], p. 311



McCormick [8], p. 62.

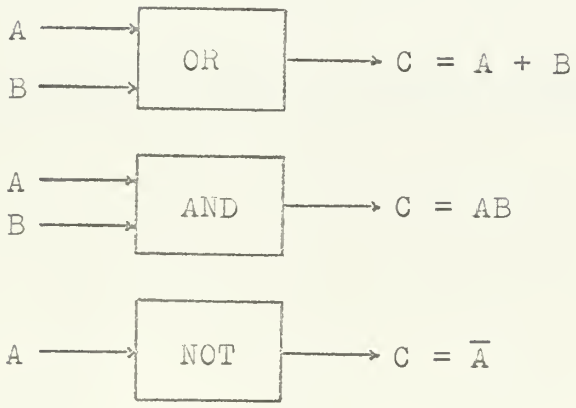
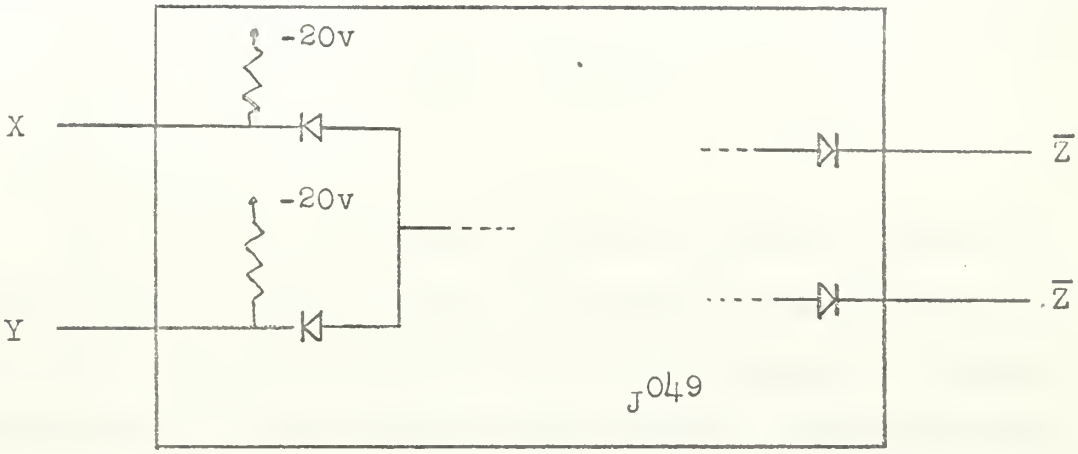


Figure 3-1 Typical Logic Symbols



$X + Y = Z$

Figure 3-2 Basic Inverter

$X + Y = Z$

Figure 3-3 Basic Inverter Symbol

will be represented by a circle on the logic diagram. Examples of this connection are shown in figures 3-4 and 3-5.

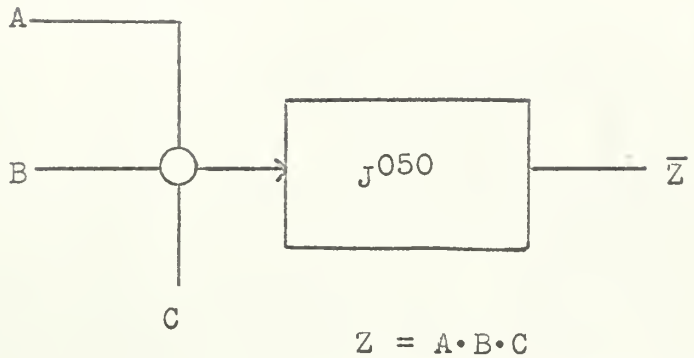
3.2 Flip-flop

The bi-stable multivibrator, flip-flop, is used as a temporary storage device in logical circuitry. The circuit is formed by the interconnection of two inverter circuits, as shown in figure 3-6. This will be represented in a logic diagram by two inverters, placed one on top of the other to form a square, as shown in figure 3-7.

To demonstrate the storage property of the flip-flop observe what happens in the circuit of figure 3-6 when a logical "1" is applied at the set input. The output of K^{000} will be a "0", and this is the input to K^{001}. A "0" at the input to K^{001} will be inverted to a "1" at the output. This "1" is fed back to the input of K^{000}, and is also the set output of the flip-flop. Now the "1" at the set input can be removed and the state of the flip-flop will not change until it is cleared.

To clear the FF a "1" is applied at the CLEAR input terminal. This causes the output of K^{001} to go to a "0", and thus the input to K^{000} to go to a "0". This input "0" will be inverted to a "1" by K^{000}, and this "1" is fed back to hold down K^{001}. The FF will now remain in the cleared state, a "0" out at the upper terminal and a "1" out at the lower terminal, until it is again SET.

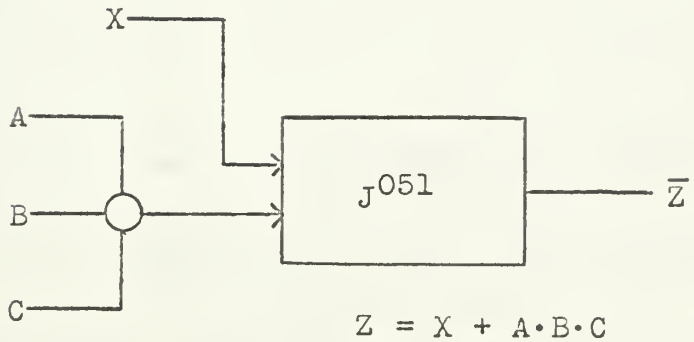
Two rules will be followed in assigning inverter numbers to a FF. These are (1) that the two assigned designators will be consecutive numbers, and (2) that the even numbered designator will always be assigned to the upper inverter. Referring to figure 3-6, the SET input will always be the input to an even numbered inverter; the SET output will always come from an odd numbered inverter.



$$Z = A \cdot B \cdot C$$

$$\overline{Z} = \overline{A \cdot B \cdot C} = \overline{A} + \overline{B} + \overline{C}$$

Figure 3-4 Logic Diagram



$$Z = X + A \cdot B \cdot C$$

$$\overline{Z} = \overline{X + A \cdot B \cdot C} = \overline{X} \cdot \overline{A \cdot B \cdot C}$$

$$= \overline{X} \cdot \overline{A} + \overline{B} + \overline{C}$$

$$= \overline{A} \cdot \overline{X} + \overline{B} \cdot \overline{X} + \overline{C} \cdot \overline{X}$$

Figure 3-5 Logic Diagram

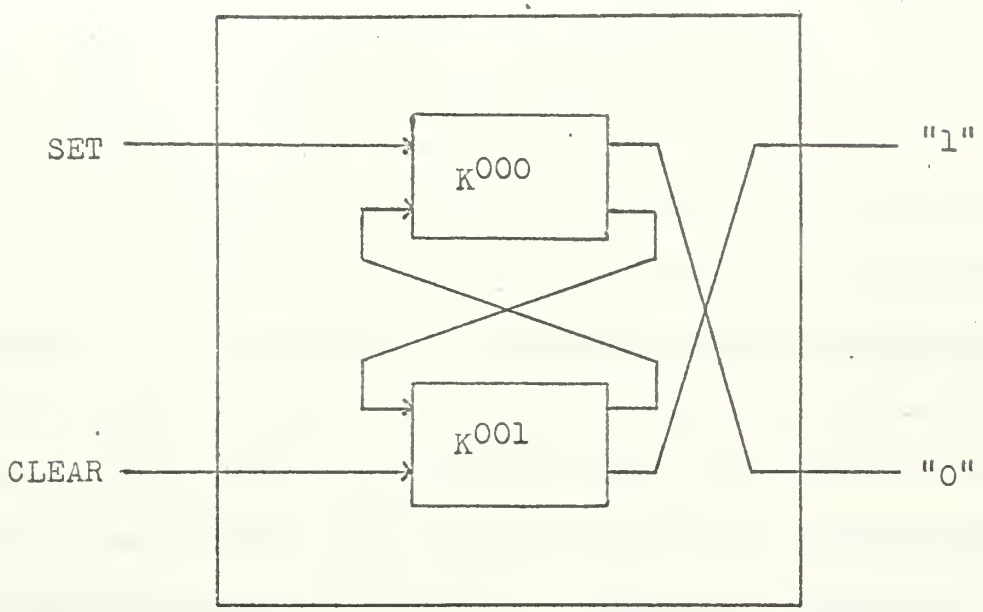


Figure 3-6 Interconnection of Inverters to Form a Flip-flop

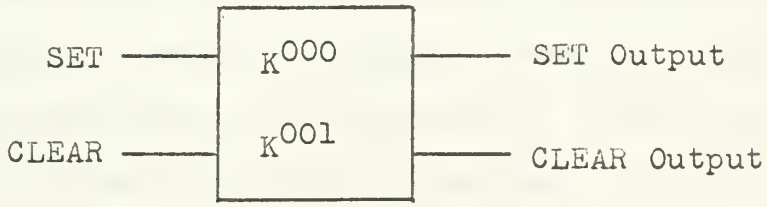


Figure 3-7 Logic Diagram Symbol for a Flip-flop

3.3 Solution of Design Problems using the Logic Diagram

The logic symbols developed in sections 3.1 and 3.2 are particularly useful in the design of unclocked logical circuitry. Since the digital "interface" (DIF) mentioned in the introduction uses unclocked logic, its design will be used as the source of two examples. The examples selected involve the decoding of functions received from the computer, and the handling of a sequence of program instructions. These examples have been selected because they illustrate how a designer can use the basic logic symbols to obtain the complete solution to a design problem. It is important to note that a finished logic diagram which represents a solution is a complete expression of the circuitry which will be required to build the logic.

3.3.1 The Design of the Decoding Circuitry

The digital "interface" (DIF) controls the transfer of data between a CDC 1604 computer, a CDC 160 computer, or between either of these computers and a radar data transfer and operators control consol. The DIF operates under the program control of both computers. Therefore it is required to recognize instructions from these computers, and to take appropriate action on these instructions. In this section the design of the decoding circuitry will be described, and in the following section the design of the circuitry to act on a specific program will be discussed. In the discussion of each of these design problems emphasis will be placed on the use of the logic diagram to obtain the solution.

The program instructions which will be used by the 1604 computer to control the digital "interface" are listed in table 3-1. The finished

design of the decoding circuitry is shown in figure 3-8. To understand how this design was accomplished consider a specific instruction, say 74.0.65000, Select Interrupt on the 160 Ready to Receive Data from the 1604. The lower order 12 bits of this instruction will come into the DIF on the function code lines shown at the left of figure 3-8. These lines come into the M^{XXX} inverters. These M inverters are special input circuits, not discussed in section 2, which convert an external logical "1" (-0.5v) received at the input to the inverter to an internal logical "1" (-3.0v) at the output of the inverter. This is illustrated in figure 3-9. For the purpose of this discussion it suffices to remember that a logical "1" into an M^{XXX} results in a logical "1" out, a "0" in results in a "0" out. The upper four octal digits of the instruction 74.0.65000 are not sent to the DIF but are decoded within the 1604 and cause the computer to send out a "1" on the Output Function Ready line from the 1604. This is shown at the bottom of figure 3-8. A 74.0.5XXXX instruction would have caused the computer to send out a "1" on the Input Function Ready line.

The inputs to our design problem then are the bits on the 12 function lines and either the Input Function Ready signal or the Output Function Ready signal. The desired result required from the design is to SET the flip-flops at the right of figure 3-8. For the specific instruction considered, 74.0.65000, the desired result is a "1" at the input to K^{408}. Since for this instruction there is a "1" on the Output Function Ready line this can be used to gate a "1" into K^{408} to SET the FF. A "1" can be obtained at the output of an inverter, in this case J^{400}, by selecting inputs which are all "0". These "0"s for the lowest

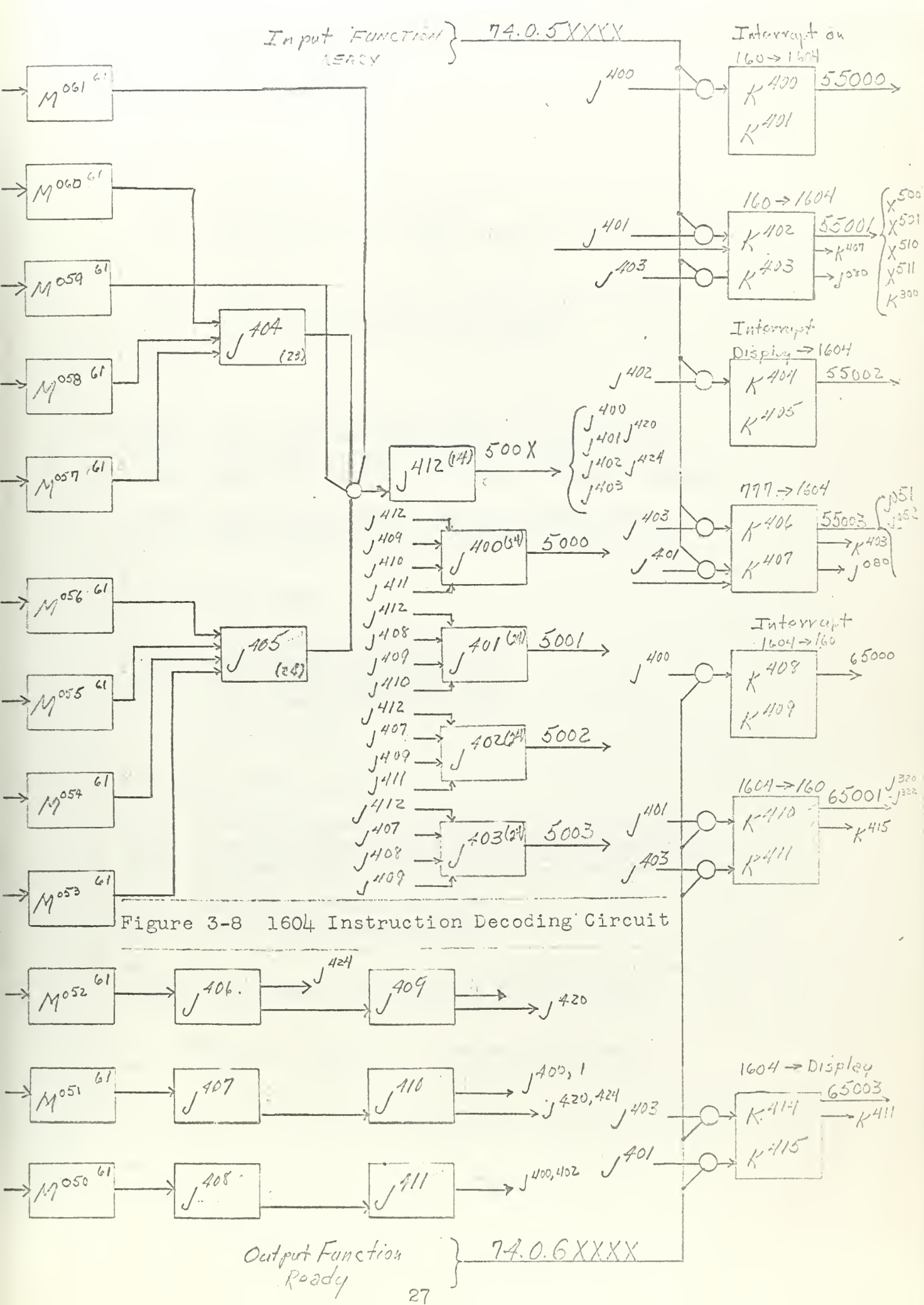


Figure 3-8 1604 Instruction Decoding Circuit

1604 Instruction Codes

SELECT (74.0.)

55000	Interrupt on 160 ready with data for input to 1604
55001	Select buffer input from 160
55002	Interrupt on OCC* ready with data for input to 1604
55003	Select buffer input, OCC* to 1604
65000	Interrupt to 160 ready to receive data from 1604
65001	Select buffer output 1604 to 160
65003	Select buffer output, 1604 to OCC*

SENSE (74.7.)

55000	Exit on 160 ready with input data for 1604
55001	Exit on 160 not ready with input data for 1604
55004	Exit on OCC ready with data for input to 1604
55005	Exit on OCC not ready with data for input to 1604
65000	Exit on 160 ready to receive data from 1604
65001	Exit on 160 not ready to receive data from 1604

160 Function Codes

5001	Select output from 1604 to 160
5002	Select input to 1604 from 160
5004	Select output from OCC* to 160
5010	Select input to OCC* from 160
5020	Request status

160 Request Status Response

0001	Interrupt request from 1604 for 160 output to 1604
0002	Interrupt request from 1604 for 160 input from 1604
0004	Interrupt request from 1604 for OCC* ready with input for 1604
0010	1604 in output mode to OCC*
0020	1604 in input mode from OCC*

*Radar Data Transfer and Operator's Control Console

Figure 3-9 Table 3-1, Program Instructions

order octal digit of the function code can be obtained at the outputs of J^{409}, J^{410}, and J^{411}. Since the next three higher order octal digits of the function code are the same for all of the instructions, 500X, these can all be decoded together. One convenient way of doing this will be to decode for a "0" which can be ORed with the decoding of the lowest order octal digit of the function code. Thus the desired output of J^{412} will be a "0", the input a "1". To obtain this "1" it is only necessary to invert the bits of the 500 which are "0" and AND these outputs with the bits of the function code which are "1".

Thus the design problem is solved right on the logic diagram by choosing a proper combination of logic circuits which will express the desired result in terms of the input parameters of the problem. It is certainly true that the problem could have been solved using some other symbolism, such as a or a , but the beauty of choosing a symbol which represents the actual basic circuit should now be apparent. The solution which results on the logic diagram actually represents the required circuitry. Of course for this to hold the rules given in section 2.3 must be respected. For example, consider the selection of J^{404} and J^{405} to collect the "0"s of the 500 and invert these to a "1". If there were no restrictions on the number of inputs one inverter could have been used to accomplish this task. Since there is no inverter with more than six OR inputs, two inverters must be used; one, J^{404}, will be on a 23 card, and the other, J^{404}, will be on a 24 card.

The logic circuitry for decoding the 160 function code is shown in figure 3-10. In the case of the 160 the function code comes out over

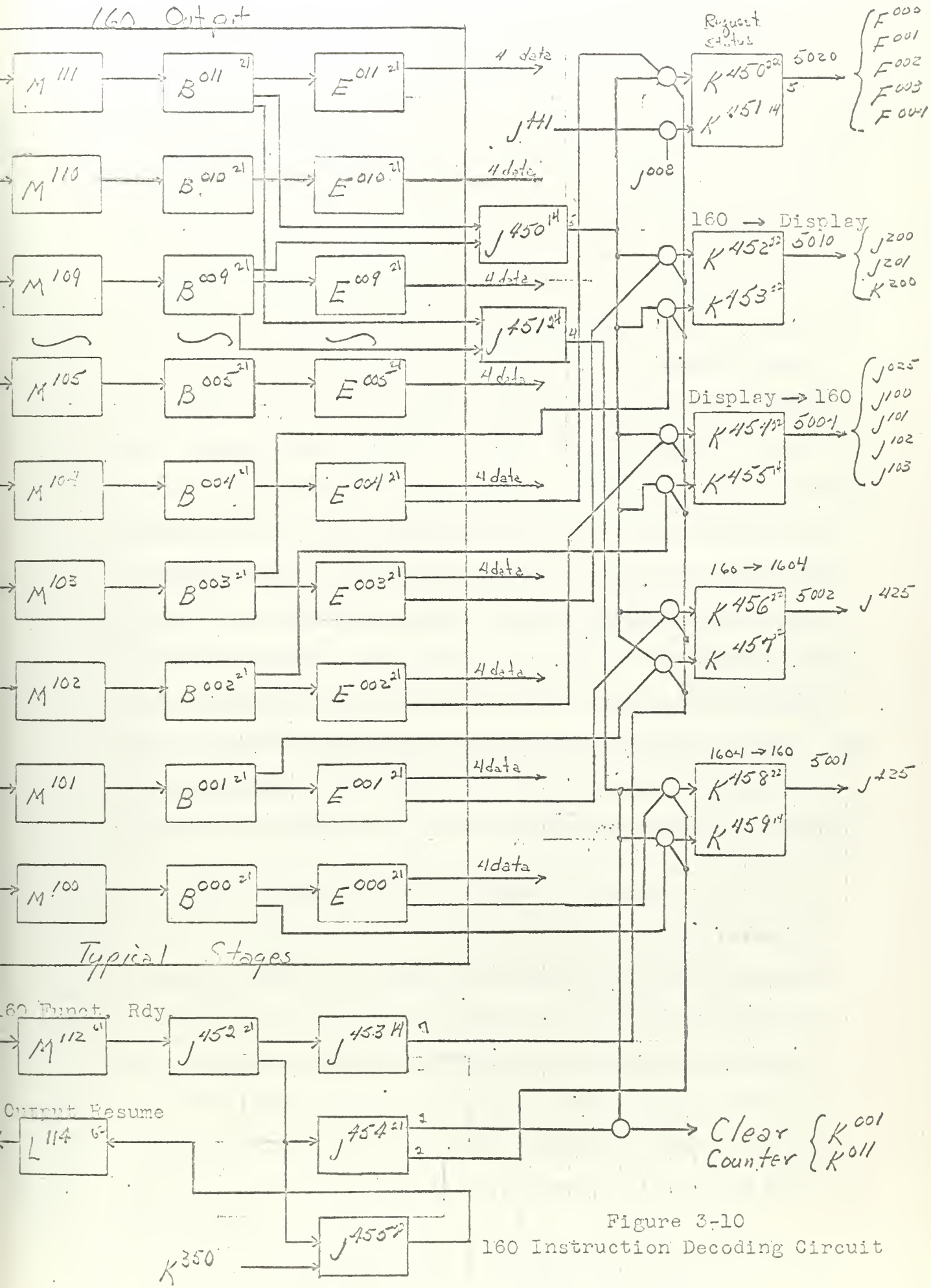


Figure 3-10
160 Instruction Decoding Circuit

the same output cable that carries the data, and a "1" is placed on the function ready line to indicate that the data on the output cable is to be interpreted as a function code. The design problem for decoding the 160 function code is much simpler than the problem encountered in the design of the 1604 decoding circuitry. This is because a two out of six code is used in selecting the coding of the upper six bits in the 160 function code. Thus there are only $\binom{6}{2}$ or 15 possible codes available which can be assigned to these octal digits. The code 50 was selected, and assigned to the DIF. This code then will have "1"s in the first and third bit positions, 101 000, and those lines which contain the "1"s are the only ones which need to be decoded by the designer in order to assure that the function code on the output cable of the 160 is intended for the DIF. A one bit coding scheme was selected for the lower order six bits. Thus only one of these bits needs to be decoded to set a particular flip-flop. Here again this design was accomplished right within the framework of the logic diagram. The decoding scheme for the 160 word is such that every function code assigned to the DIF will SET one of the flip-flops and clear all the others.

3.3.2 The Execution of an Instruction Sequence

Now the same technique will be employed to design the circuitry to carry out a series of instructions from the two computers. In particular the problem to be discussed is the design of the circuitry necessary to establish an output buffer link from the 1604 to the 160. The 1604 will start the sequence of instructions by selecting an interrupt on the 160 ready to receive data from the 1604, 74.0.65000. This instruction will be decoded as previously described and $K^{408/409}$ will be SET.

In the course of its program the 160 will execute a Request Status instruction, 5020, which will be decoded by the circuitry of figure 3-10 to SET $K^{450/451}$. The next instruction in the 160 program following the Request Status will be a one word input instruction. This one word will be the response to the Request Status. Therefore it is necessary for the DIF to make up a one word response and make it available at the input cable to the 160. This response will then be gated into the 160 when the input instruction is received. The response which the DIF must put on the input cable is 0002, to indicate that the 1604 has selected an interrupt when the 160 is ready to receive data from the 1604. This will require the setting of just one bit on the output cable. This can be done quite simply by just ANDing the SET output of the Request Status FF with the SET output of the Interrupt FF, $K^{408/409}$. This is shown in figure 3-11. F^{001} is an inverter in line with the second bit of the input cable to the 160. The 160 computer will be programmed to interpret this response and select an input from the 1604, 5001. This function code will be decoded by the DIF and $K^{458/459}$ will be SET. This action of the 160 satisfies the condition on which the 1604 originally requested an interrupt. This interrupt response is sent to the 1604 by ANDing the SET outputs of K^{409} and K^{459}, see figure 3-12.

When the interrupt response signal is received by the 1604 the program will be interrupted and will jump into a subroutine to sense the source of the interrupt. One of the instructions in this subroutine will be a 74.7.65000 or a 74.7.65001. The lowest order bit is not transmitted by the 1604 and therefore it is not decoded by the DIF.

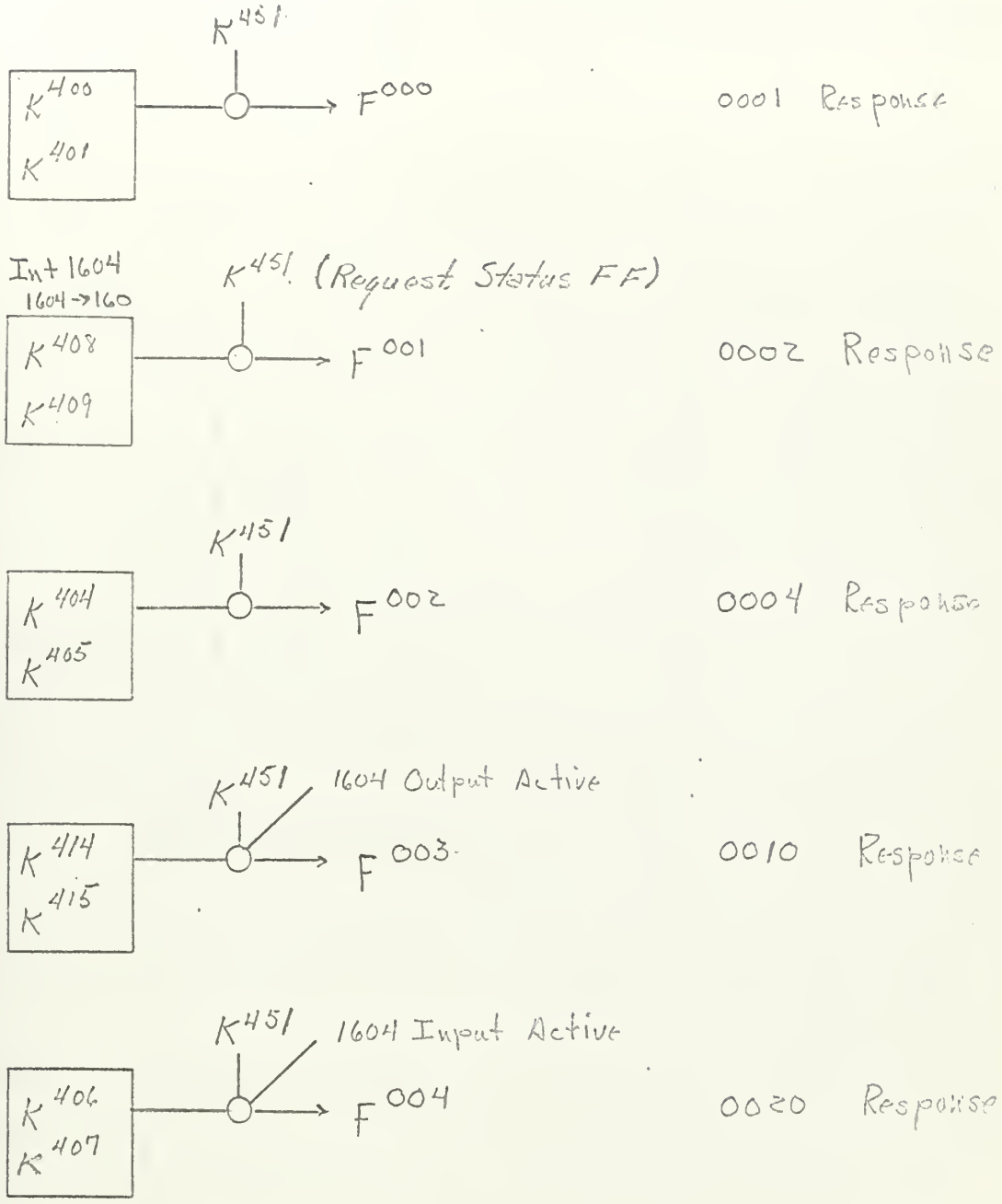
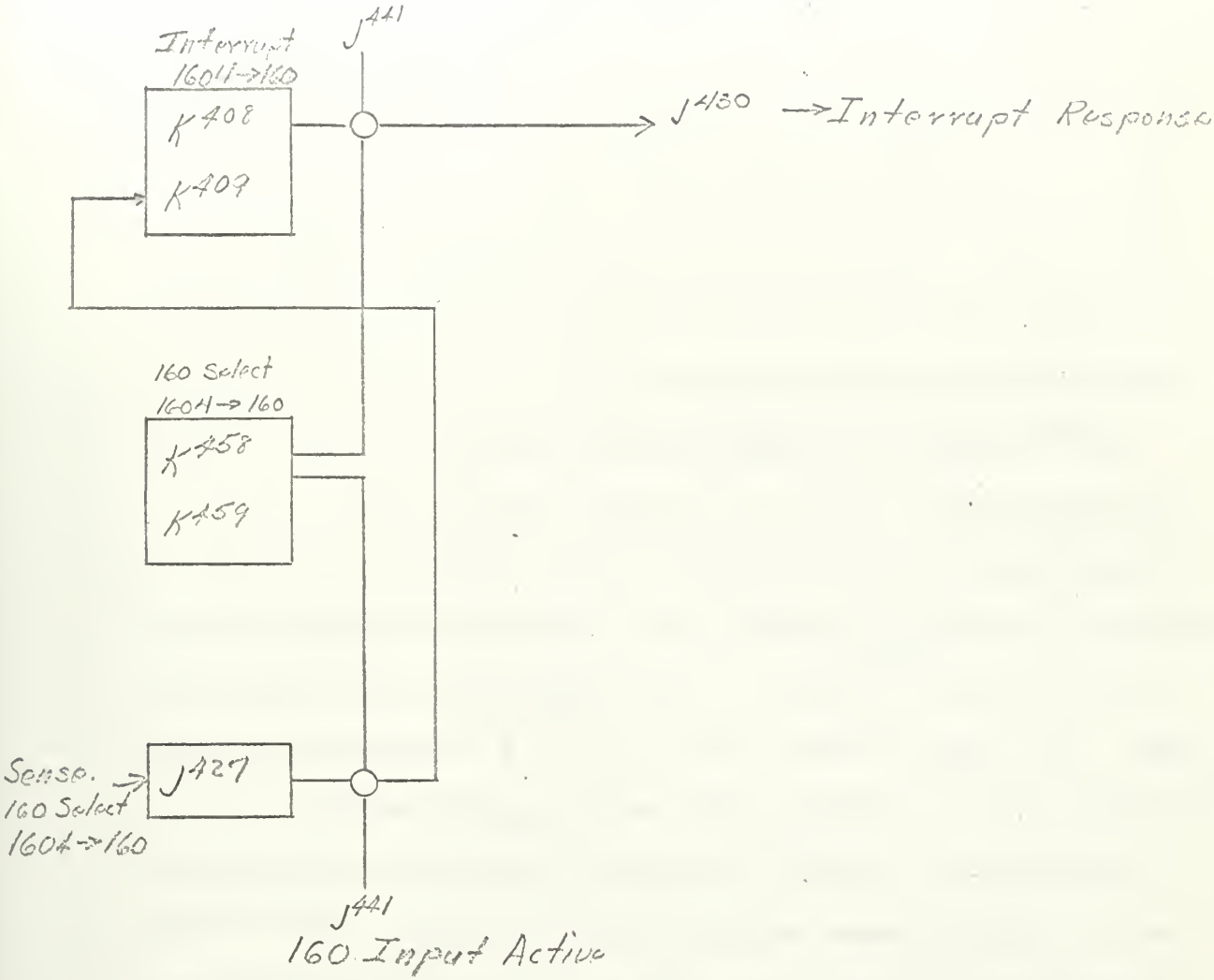


Figure 3-11 160 Request Status Response



J441, 160 Input Active

J430, Interrupt Response

Figure 3-12 1604 Interrupt Response

The other bits of the 5000/1 are decoded by the function decoding circuits of figure 3-8, but this time the outputs from J^{409}, J^{410}, and J^{412} will be ORed with a NOT Output Sense Ready signal at the input to J^{427}. This is shown in figure 3-13. No FF is set in decoding the SENSE instructions since they are held on the function lines for 8 microseconds, and it is during this period that the 1604 samples the Sense Response line. The Sense Response will be a "1" since K^{459} is a "1" and the 160 Input Active signal is up. This "1" Sense Response is combined in the 1604 with the lowest order bit of the instruction to determine whether the program does a half-exit or a full-exit to satisfy the specific Sense instruction. The "1" out of J^{427} is also used to clear the Interrupt FF, $K^{408/409}$. This is shown in figure 3-12. Upon receipt of the Sense Response the 1604 will select an output to the 160 by executing a 74.0.65001 instruction. This will be decoded and $K^{410/411}$ SET. This will activate the gates shown in figure 3-14 for the buffer of data from the 1604 to the 160. The X^{xxx} inputs shown in figure 3-14 are from a counter which is used to divide the 48 bit 1604 word up into four 12 bit words for input to the 160. This is necessary since the bit length of the 160 data word is only 12 bits. Thus only 12 of the data gates in figure 3-12 will be SET at any one time. This completes the program sequence necessary to establish an output buffer link from the 1604 to the 160.

Here again the parameters of this problem were particularly well suited to a direct solution using the logic diagram. The logic diagram can also be used to solve problems where the parameters are expressed by Boolean equations. However the direct translation of a Boolean

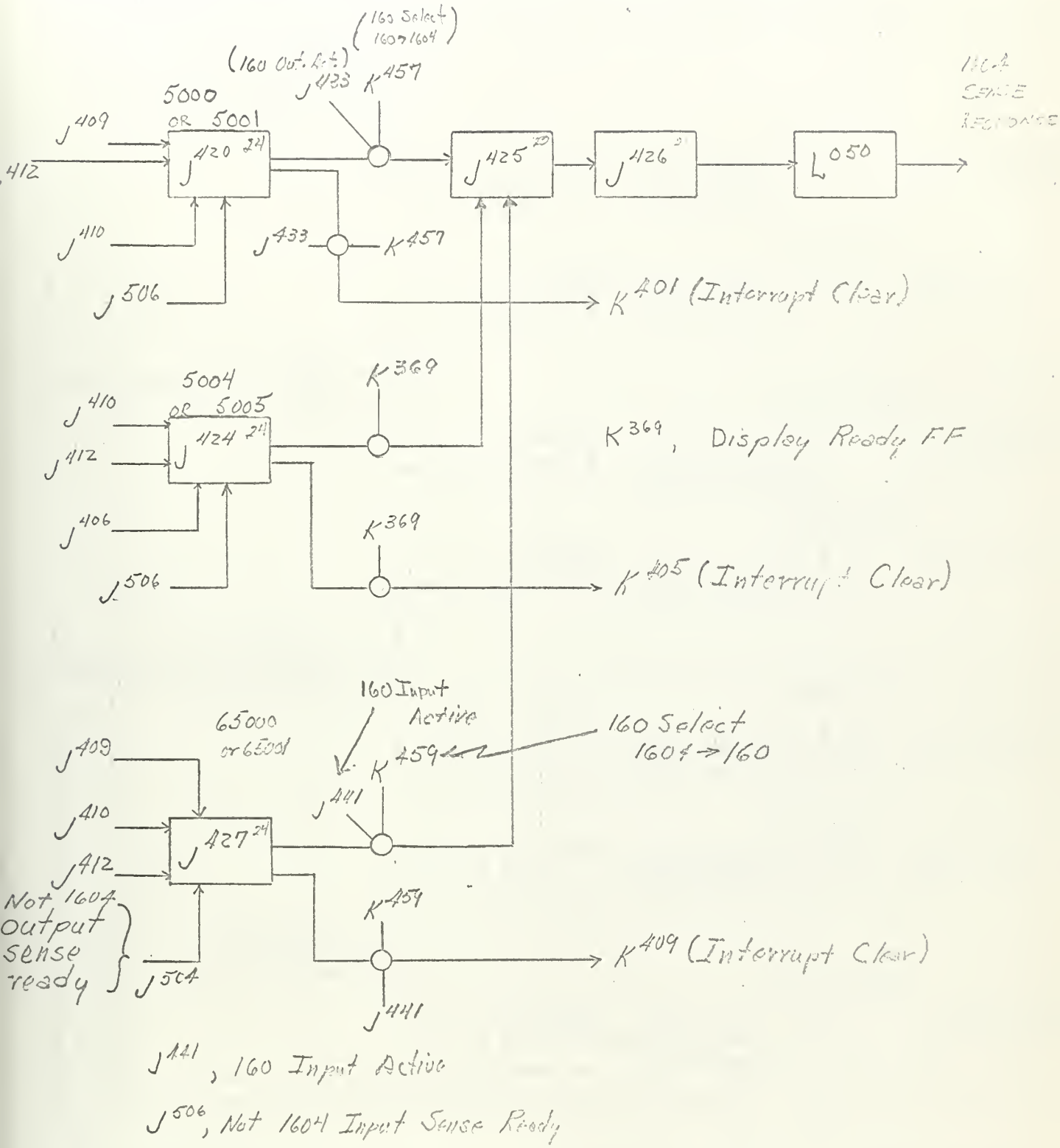


Figure 3-13 1604 Sense Response

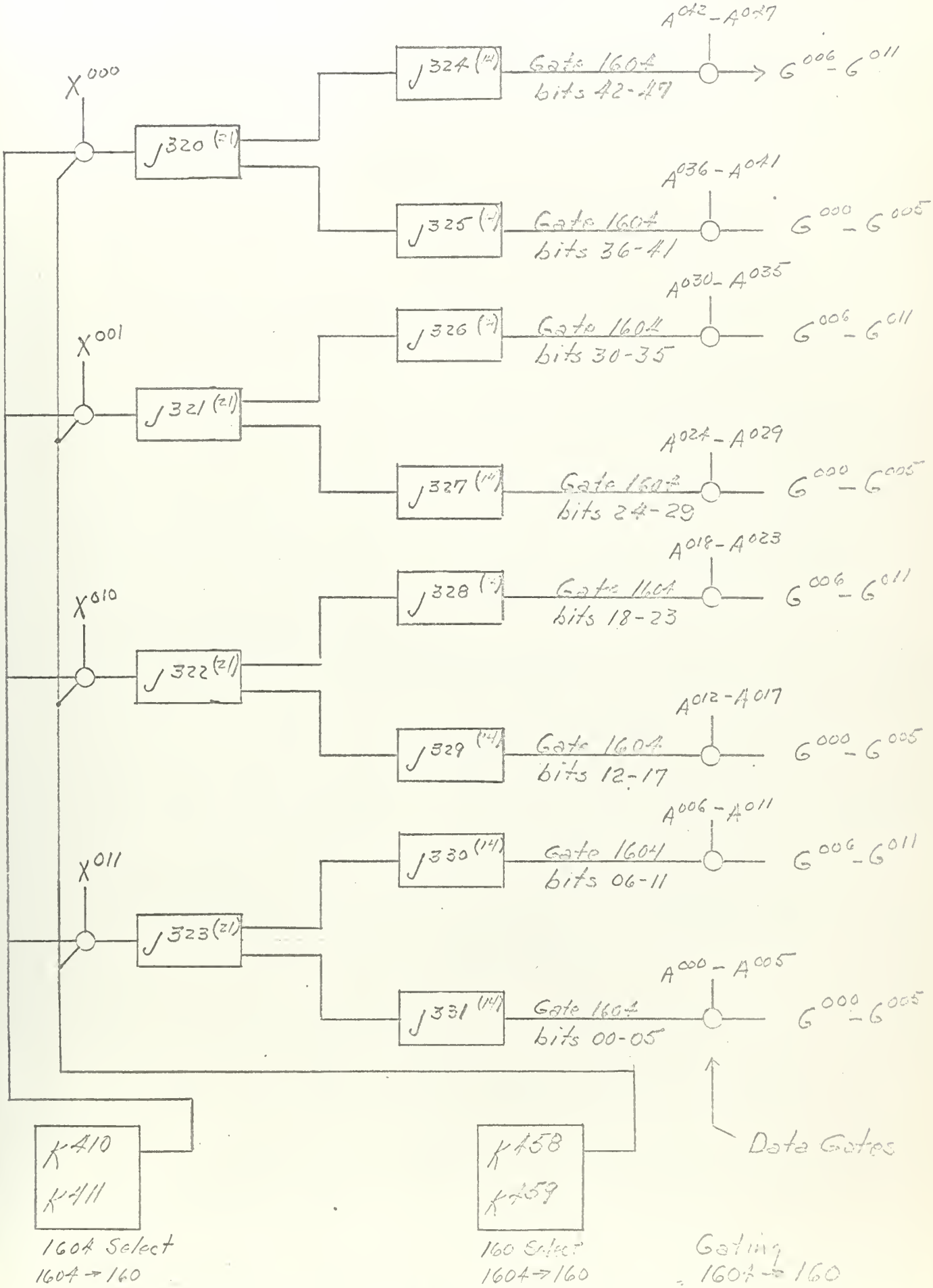


Figure 3-14 Gating 1604 Data into 160

equation into a logic diagram may not lead to an optimum solution which uses a minimum number of inverter circuits. This problem will be discussed in the next section.

4. Derivation of an Optimum Boolean Expression

In the preceding section the logic diagram was used directly to gain the solution to a design problem. The example given was particularly well suited for this technique. However in problems where the design criteria are expressed in the form of Boolean equations this direct approach may not lead to an optimum solution. This results from the fact that a Boolean equation can be expressed in many forms, i.e., a sum of products, a product of sums, or other equivalent forms.[1] Since any design is to ultimately result in circuitry which is constrained to a basic logic circuit, it seems probable that one form of logical expression may result in the simplest circuitry. The following example will provide an intuitive answer to this question of form.

4.1 The Simple Binary Adder

The truth table for binary addition is

A	1	1	0	0
B	1	0	1	0
Sum	0	1	1	0
Carry	1	0	0	0

One form of the Boolean equation for the sum and the carry is

$$(1) \qquad S = (A + B) \cdot \overline{(A \cdot B)}$$

$$(2) \qquad C = A \cdot B$$

The logic diagram which obtains from (1) and (2) is shown in figure 4-1. It will take seven inverters to build the circuit of equations (1) and (2).

[1]For a thorough discussion of the various forms of a logical expression see Ledley /7/, p. 324.

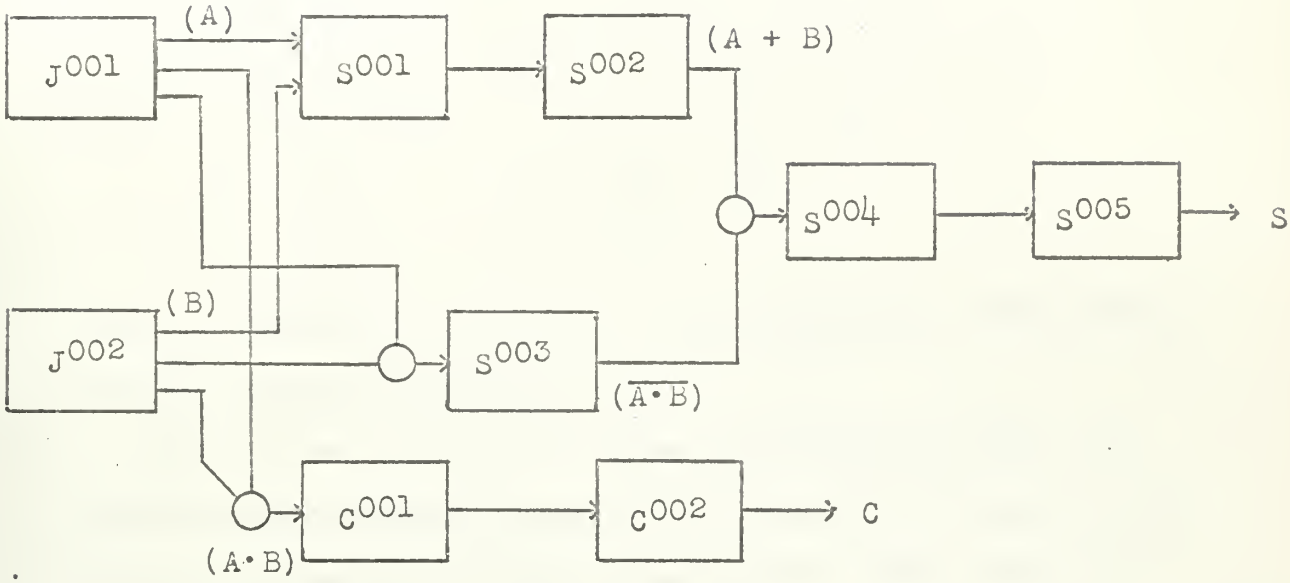


Figure 4-1 Binary Addition with Seven Inverters

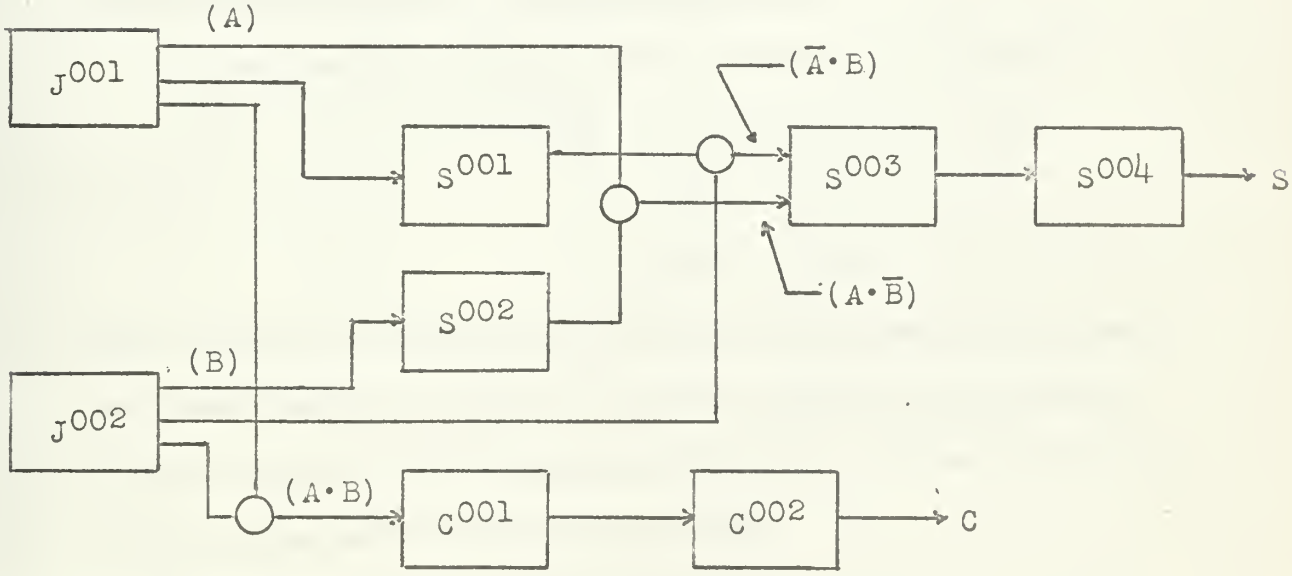


Figure 4-2 Binary Addition with Six Inverters

Applying De Morgan's rule; Appendix A, equation 9; to equation (1)

$$S = A + B \cdot \overline{(A \cdot B)} = (A + B) \cdot (\overline{A} + \overline{B})$$

$$= A(\overline{A} + \overline{B}) + B(\overline{A} + \overline{B})$$

$$= 0 + A\overline{B} + B\overline{A} + 0$$

$$(3) \qquad S = A\overline{B} + \overline{A}B$$

$$(4) \qquad C = A \cdot B$$

The logic diagram for (3) and (4) is shown in figure 4-2. This expression will take six inverters to build.

The problem in each logic diagram is that after the AND's and OR's have been wired the answer is in inverted form, $\overline{S}$ and $\overline{C}$; and additional inverters are required to get S and C. This results, of course, from the fact that the AND and OR logic is wired at the input to an inverter card, and any logical sum which results will of necessity become inverted before it reaches an output pin.

To overcome this difficulty solve (1) and (2) for $\overline{S}$ and $\overline{C}$.

$$\overline{S} = \overline{A + B \cdot \overline{(A \cdot B)}} = \overline{(A + B)} + (A \cdot B)$$

$$(5) \qquad \overline{S} = \overline{A} \cdot \overline{B} + A \cdot B$$

$$(6) \qquad \overline{C} = \overline{A \cdot B} = \overline{A} + \overline{B}$$

Since this logic is connected at the input of an inverter the output of the inverter will be S and C. The logic diagram for (5) and (6) is shown in figure 4-3 and indicates that only four inverters will be required to generate the binary sum and carry.

A conclusion that can be drawn from this brief example is that fewer inverter cards will be required if the form of the desired equation is the logical sum of the products in an equation for the NOT of

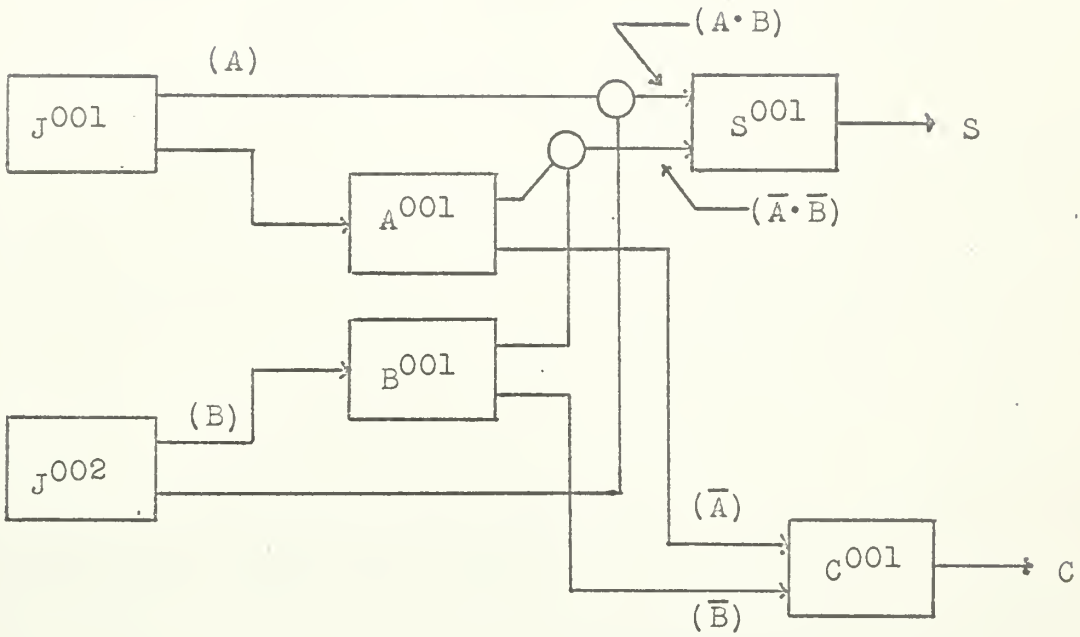


Figure 4-3 Binary Addition with Four Inverters

the desired result. This conclusion should not seem too strange since this form clearly represents the logic of a single inverter circuit.

5. Transcription of the Logical Design into Wiring Instructions

The chassis wiring of even a simple logical design can become a very complicated problem. To overcome the complexity and reduce the chance for errors a three step process will be described which has as its end result the production of a wire tabulation. This tabulation contains the origin and destination of every wire on the chassis. With it as a guide the wiring of the chassis becomes a routine matter.

5.1 File of Equations

The first step in the process is the development of a file of equations which describe the logic of the design. The format of the file of equations is illustrated in figure 5-1. Each entry in the file consists of two lines. The first line is a Boolean equation for one inverter. The entries on the right-hand side of the equation are the OR inputs to the inverter whose number appears on the left-hand side of the equation. Each of the OR entries may consist of several AND terms. Since each of the equations in the file represents an inverter, it is understood that the term on the left of the equals sign is the logical NOT that results from the Boolean relation on the right. That is, the equation of figure 5-1 might better be written

$$\overline{K^{000}} = K^{001} + E^{000}X^{111} + J^{062}0^{000} .$$

If any of the OR terms on the right are a logical "1", the output of K^{000} will be "0". Since this relation holds for every inverter in the file of equations, the line is not normally written above the term on the left of the equals sign, but is understood. On the second line of the format there are three entries. The leftmost entry is the position

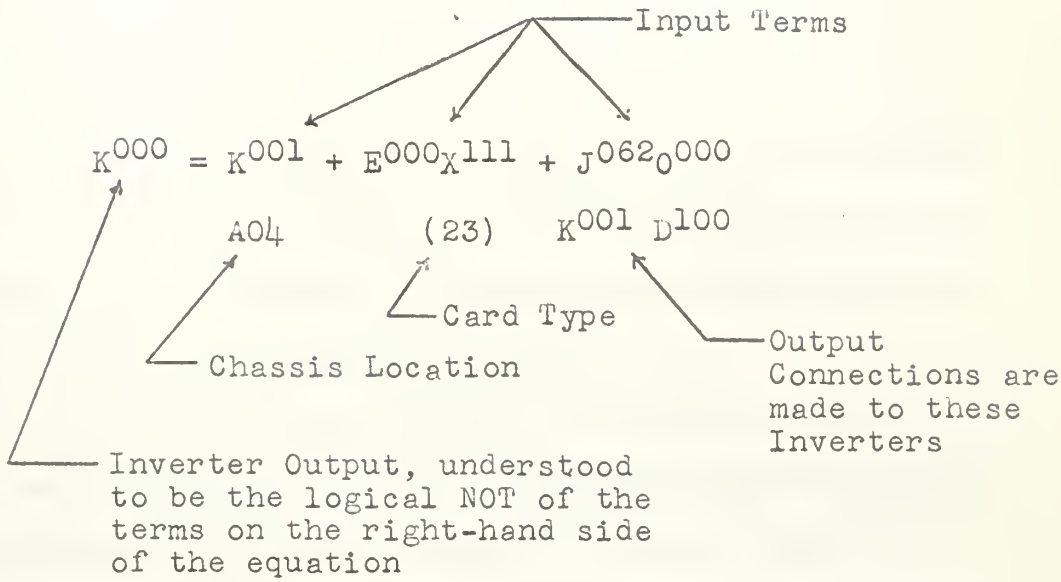


Figure 5-1 Format of Equation File

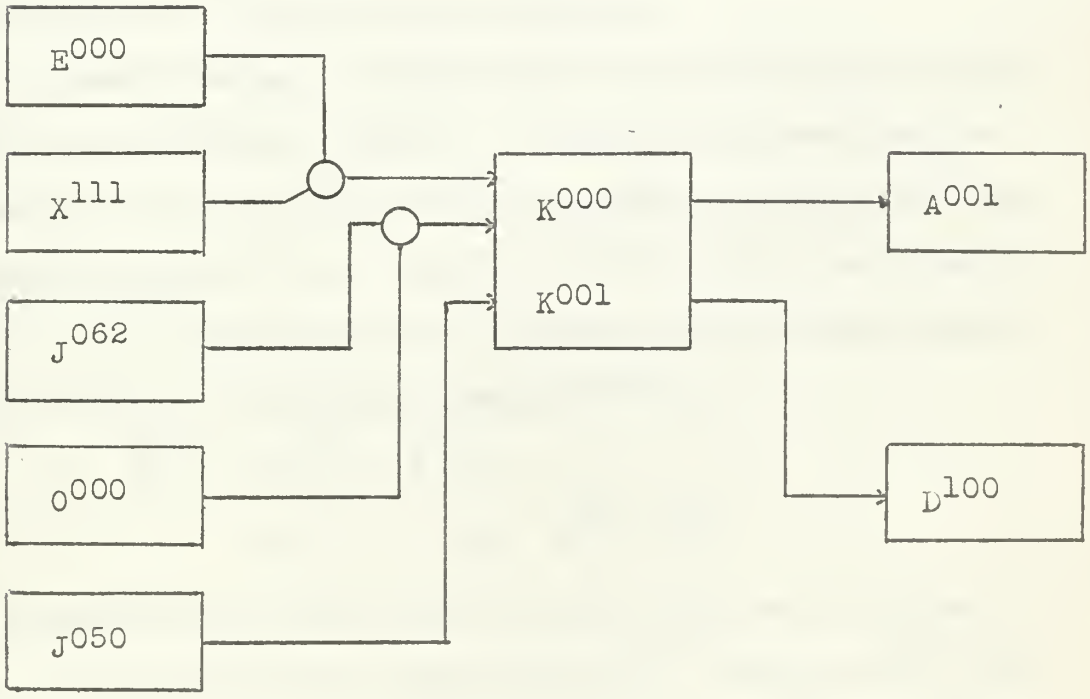


Figure 5-2 Example Logic Diagram

on the chassis of the logic card which contains the subject inverter. More information about this entry will be given in the next section. The entry in parenthesis on the second line is the type of logic card on which the inverter is mounted. The inverters which are connected to the output of the subject inverter are listed in the right-hand entry of the second line. An example will illustrate the technique of going from the logic diagram to the file of equations.

Consider the logic diagram of figure 5-2 and write the file of equation entries for the FF, $K^{000/001}$. Recall that the FF square on a logic diagram represents two inverters with internal feedback and crossed output leads, see figure 3-6. Here the term internal feedback may be misleading. It is used to mean that the connections between the inverters are not shown on the logic diagram, but they must most surely be wired in the circuit by the designer.

Comparing the logic diagram of figure 5-2 and the logic equation of the previous example, figure 5-1, it should be apparent that the equation is written for K^{000} in the logic diagram. Note that K^{000} has an output, not shown in the diagram, to K^{001}; and the other output is to D^{100}. K^{000} can be built using a 23 card since it has three inputs and two outputs. The equation entry for K^{001} is

$$K^{001} = K^{000} + J^{050}$$

$$A04 \qquad (23) \qquad K^{000} \; A^{001}$$

The feedback term is listed first on the right-hand side of the equation for each half of a FF. In the equation K^{001} outputs to K^{000} and to A^{001}. K^{001} could have been constructed using a 22 card but since the other half of the 23 card for K^{000} is available this was used. Thus

both entries in the file have the same chassis location number, A04.

The rules governing the use of the CDC logic card were given in section 2 3. In light of these rules certain conditions can be placed on the file of equations. These are:

(1) No more than eight output terms can be listed in the second line.

(2) A maximum of six OR terms can appear on the righthand side of an equation.

(3) No OR term can contain more than four AND terms.

(4) The total number of OR terms and output terms cannot exceed 12.

5.2 Chassis Map

The chassis map is a diagram which shows the location of each inverter on the chassis. The diagram also shows the type of card which is to be installed in each location. A typical chassis map is shown in figure 5-3. It is a block drawing of each card location from the card side of the chassis. The rows on the drawing are identified by letter; A for the top row, B for the second row, etc. The card locations in a row are numbered from left to right; the left-most card is number 1.

The chassis map is used by the designer to lay out the card locations to achieve a minimum wire run. When a layout has been decided upon, the card location numbers, A01, B12, etc., are entered in the file of equations.

5.3 Wiring Tabulation

The wiring tabulation (wire tabs) is the final step in the logical design. This is a list of every wire which is to be connected on the chassis.

	01	02	03	04	05	06
A	E^{000} X^{111} 21	J^{062} O^{000} 21	J^{050} 14	K^{000} K^{001} 23	A^{001} D^{100} 21	
B						
C						

Figure 5-3 Chassis Map

A typical set of wiring tabs is shown in figure 5-5. The "Origin" column will contain only inputs; the "Destination" column only outputs. The "Description" terms usually lists the inverter numbers of the "Origin" and "Destination" entries. Each entry in the tabulation is written for a particular inverter input. Obviously if each input is connected to the proper output, then when all the inputs have been wired, all the outputs will also be properly wired.

The wiring tabulation is written from the file of equations. An example will illustrate the technique. Consider again the flip-flop of figure 5-2. The two file of equation entries for K^{000} and K^{001} are shown again in figure 5-4. To facilitate writing the wire tabulation input pin numbers are assigned to each OR term on the right side of the Boolean equation in the file of equations. The first OR term on the right of the equal sign is assigned the lowest numbered input pin for that inverter, the second OR term the next pin, and so on; the numbers increasing from left to right. Each output term in the second line of each entry in the file of equations is assigned an output pin The output terms are numbered from left to right in descending order of the numbered output pins. Referring again to the equations in figure 5-4, $K^{000/001}$ is a FF made up of the two inverters on the 23 card located at A04 on the chassis; the upper row, fourth card from the left. The 23 card is a two inverter card with three inputs per inverter. K^{000} is the lower numbered (A) inverter and has inputs on pins 1, 2, and 3; outputs on pins 4,5, and 6. K^{001} is the higher numbered (C) inverter and has inputs on pins 7, 8, and 9; outputs on 10, 11, and 12. Thus in the file of equations, the OR inputs K^{001},

$$K^{000} = \overset{1}{K^{001}} + \overset{2}{E^{000}X^{111}} + \overset{3}{J^{062}O^{000}}$$

A04 (23) $\overset{6}{K^{001}}$ $\overset{5}{D^{100}}$

$$K^{001} = \overset{7}{K^{000}} + \overset{8}{J^{050}}$$

A04 (23) $\overset{12}{K^{000}}$ $\overset{11}{A^{001}}$

Figure 5-4 File of Equation Entries with Wire Tab Numbers

WIRE TABULATION

Title Flip-flop in Logic Diagram of Figure 4-8

SHEET 1 OF 1

Origin	Destination	Color	Gauge	Description	Length	Chg.
A04-1	A04-12			K^{000} - K^{001}		
A04-2	A01-6			K^{000} - E^{000}		
	A01-12			K^{000} - X^{111}		
A04-3	A02-6			K^{000} - J^{062}		
	A02-12			K^{000} - O^{000}		
A04-7	A04-6			K^{001} - K^{000}		
A04-8	A03-12			K^{001} - J^{050}		
A04-9	A04-14			unassigned input, Connect to ground		

Figure 5-5 Typical Wire Tabulation

$E^{000}X^{111}$, and $J^{062}0^{000}$ are assigned to pins 1, 2, and 3; the output connections which will go to K^{001} and D^{100} are assigned to pins 6 and 5 respectively. As for K^{001}, it will receive an input from K^{000} on pin 7, and from J^{050} on pin 8; the unused input, pin 9, must be grounded. The output to K^{000} will be from pin 12, and to A^{001} from pin 11. These numbers are assigned as shown in figure 5-4. The numbers have been circled in the figure. This pin number assignment system will give the broadest latitude for handling last minute card changes with ease. If a design calls for a 23 card, but none are available; the 23 card can be replaced by a 24 card, and the only wiring change will be the addition of two wires to ground the unused input terminals.

To summarize the numbering scheme, it is this; assign the lowest number to the first input and so on up; assign the highest number to the first output and number down. This may leave room in the middle for variation in card type.

Now the wire tabulation can be written by associating the proper input and output and recording the assigned pin numbers. Again referring to figure 5-4, the first input is at A04, pin 1; so this is entered under "Origin" in the wire tabulation, figure 5-5. Referring to the K^{001} entry in the file of equations it is seen that the output from K^{001} which has been assigned to K^{000} is at pin 12. Thus the proper "Destination" for the wire that originates at A04-1 is A04-12. The second input to K^{000} is from E^{000} and X^{111}. Let us assume that E^{000} and X^{111} are on a 21 card at A01 on the chassis; E^{000} might have an output to K^{000} from pin 6, and X^{111} to K^{000} from pin 12. Since this is an AND it is necessary to connect both output pins

(A01-6, A01-12) to the input pin (A04-2). This entry is shown on lines two and three of the wire tabs. The entry is read, "Connect a wire from A04-2 to A01-6, then connect a wire from A01-6 to A01-12." The particular connecting route chosen is not important electrically. That is, the connections could have been from A04-2 to A01-12, then from A01-12 to A01-6. Usually the shortest route is chosen. The other entries in figure 5-5 are the other connections for K^{000}, and the connections for K^{001}.

6. Example Design Problem

A purely hypothetical problem will illustrate the entire design procedure. The problem will be to design and construct the logic necessary to present the truth table of binary addition on the face of an oscilloscope. The logic of binary addition was discussed in section 4. The inputs to the truth table for A and B were 00, 01, 10, and 11. But these are the outputs of a two stage binary counter. Thus a counter can be used to drive the logic circuitry of figure 4-3, and a scope synched with the counter will present the results. The counter will count pulses created using logical circuitry and the output from a square wave generator, the HP 211A. The problem in block diagram form is shown in figure 6-1.

6.1 Logic Diagram and Circuit Description

The logic diagram of the pulse forming circuits and the counter is shown in figure 6-2. The output of the HP 211A is a 100 kc square wave, 3.0 volts peak to peak, between 0 and -3.0 volts. This input voltage is constrained between the limits of -2.0 volts and -3.0 volts by the input OR circuit of J^{001}.

The pulse at the input to J^{007} is formed by ANDing the outputs of J^{001} and J^{004}. At a time when there is a "1" into J^{001}, the output of J^{001} will be "0", and the output of J^{004} will be a "1". When the input to J^{001} changes to a "0", the output of J^{001} goes to a "1" and the AND with the output of J^{004} will be made, feeding a "1" into J^{007}. The input to J^{007} will remain a "1" until the output of J^{004} goes to a "0". This is delayed for the period of time that it takes for the three inverters J^{002}, J^{003}, and J^{004} to switch. Since the time for one

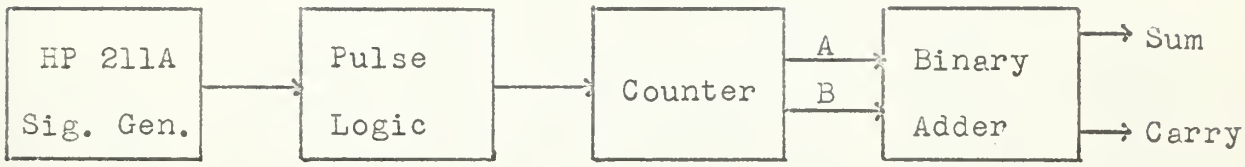


Figure 6-1 Block Diagram of Binary Adder

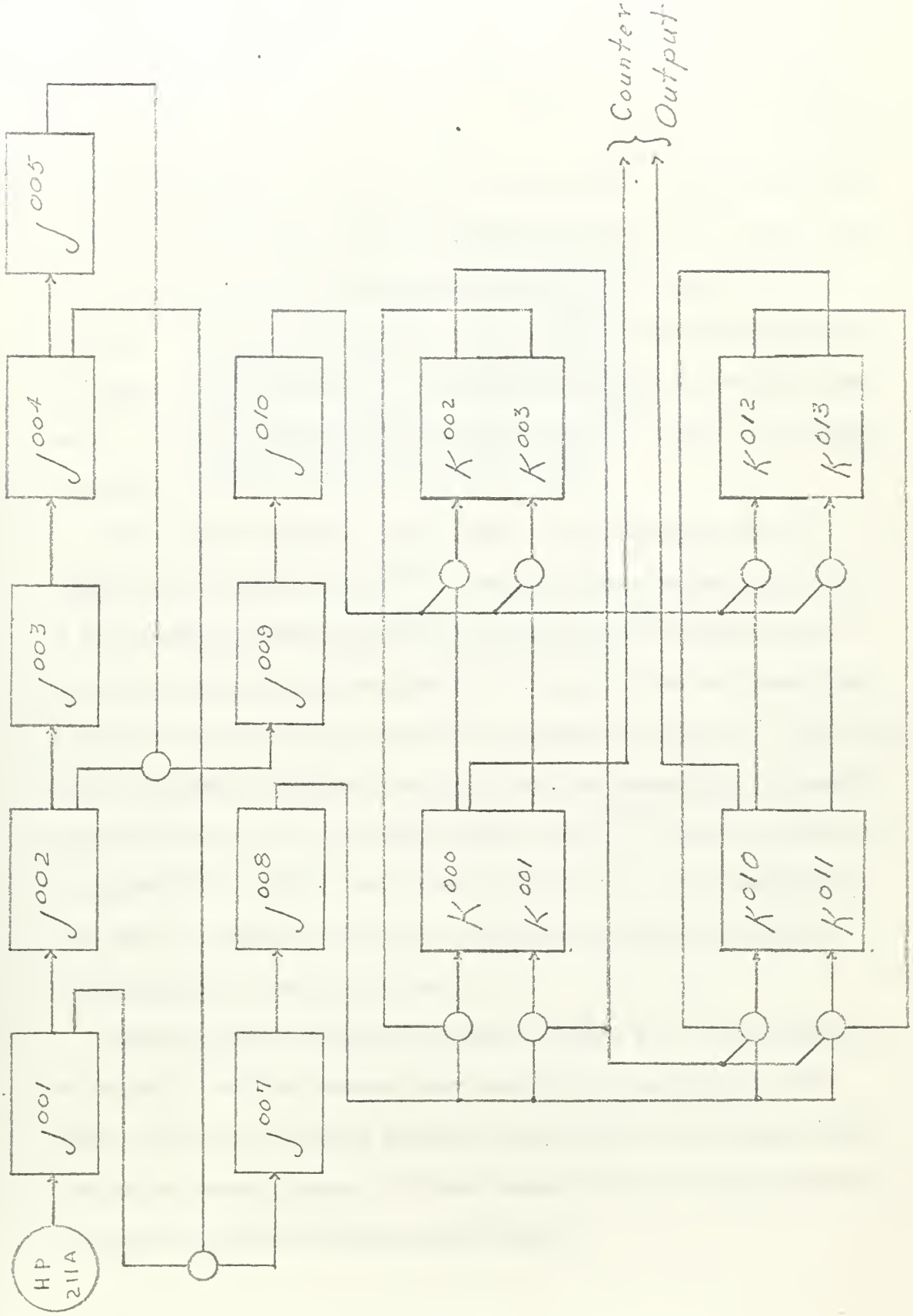


Figure 6-2 Logic Diagram of Pulse and Counter Circuits

inverter to switch is not less than 50 nanoseconds and not greater than 100 nanoseconds, the 1" input to J^{007} will last for not less than 150 nanoseconds nor more than 300 nanoseconds. The actual pulse observed at the output of J^{008} is pictured in figure 6-3. This "Fall" pulse will occur each time the input signal to J^{001} goes from a "1" to a "0". A "1" will occur at the input to J^{009} whenever the input to J^{001} goes from a "0" to a "1". This will be referred to as the "Rise" pulse. The input signal and the pulses from J^{008} and J^{010} are shown in figure 6-4.

The counter consists of two stages, and is double rank. It counts the Fall pulse from J^{008}. The count from the left-hand rank of the counter is gated into the right-hand rank by the Rise pulse. This rank readies the proper gates at the input to the left-hand rank so that the count can be increased by the next Fall pulse. The double rank is required to prevent any one pulse from advancing the counter more than one count. The output pulse from J^{008} and the counter outputs from K^{001} and K^{011} are shown in figure 6-5. It is important to note that because of the mirror in the scope camera the sweep in oscillographs is from right to left.

The adder logic diagram is shown in figure 6-6. The counter input signals, and the sum and carry outputs are also shown in the figure. The sum and carry outputs clearly agree with the truth table. The pulses which appear in the sum output are due to the differences in inverter rise and fall time in the counter.

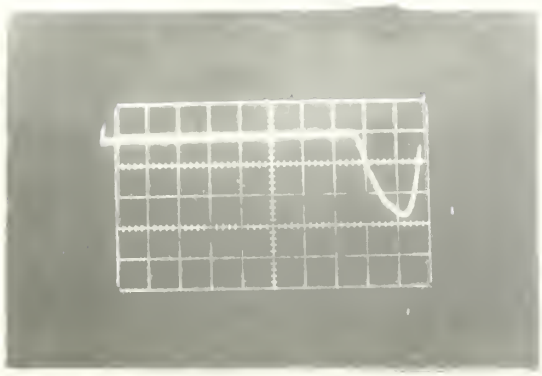


Figure 6-3 Observed Pulse at Output of J^{008}

(horizontal scale 100 nanoseconds/cm.)

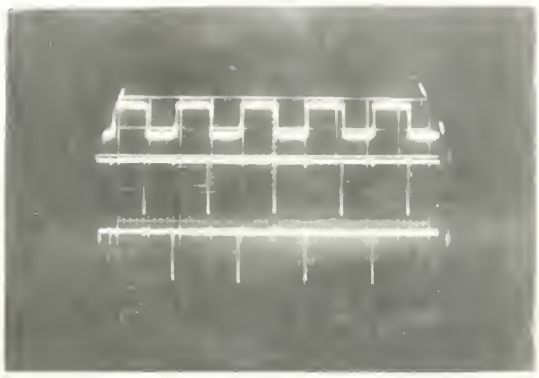


Figure 6-4 HP 211A Output and Pulses from J^{008} and J^{010}

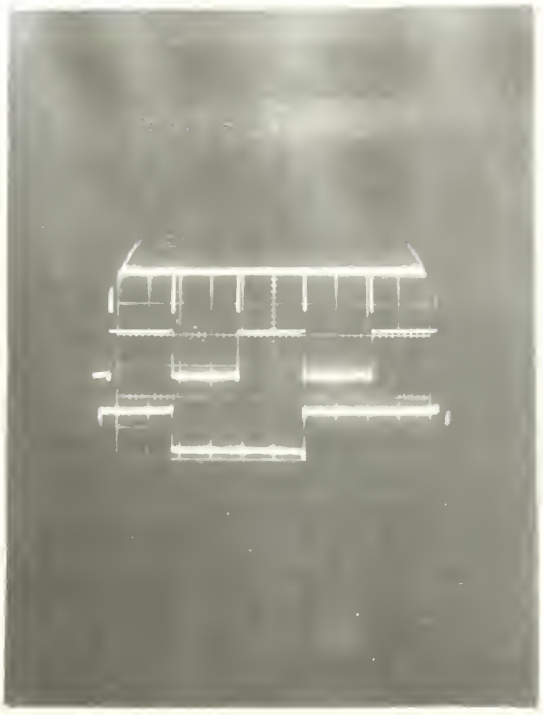
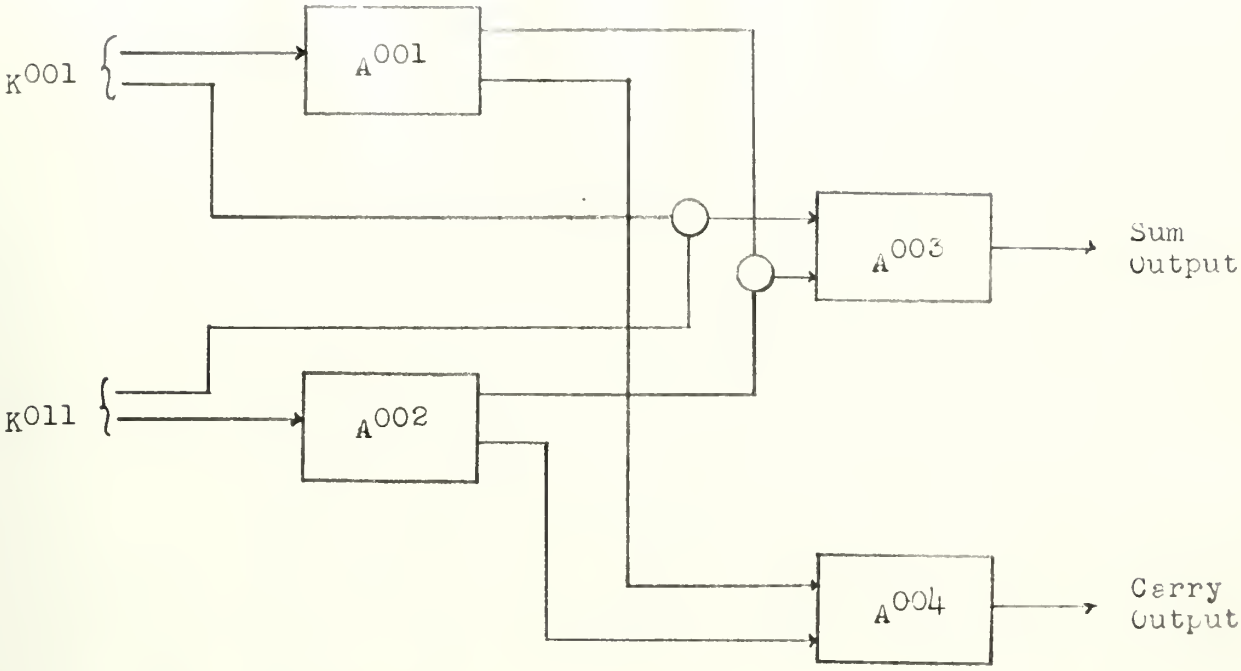
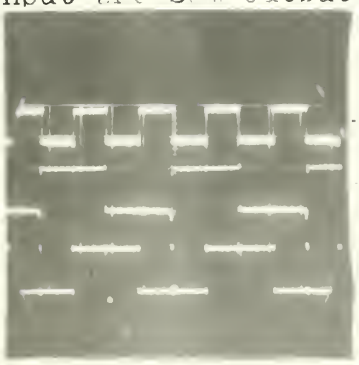


Figure 6-5 Pulse from J^{006} and Counter Output



Input and Sum Output



Input and Carry Output

Figure 6-6 Adder Logic Diagram

6.2 File of Equations

Pulse Circuitry:

J^{001} = Input from signal generator

A02 (21) J^{002} J^{007}

$J^{002} = J^{001}$

A02 (21) J^{003} J^{009}

$J^{003} = J^{002}$

A03 (21) J^{004}

$J^{004} = J^{003}$

A03 (21) J^{005} J^{007}

$J^{005} = J^{004}$

A04 (21) J^{006} J^{009}

$J^{007} = J^{001} J^{004}$

B02 (21) J^{008}

$J^{008} = J^{007}$

B02 (21) K^{000} K^{001} K^{010} K^{011}

$J^{009} = J^{002} J^{005}$

B03 (21) J^{010}

$J^{010} = J^{009}$

B03 (21) K^{002} K^{003} K^{012} K^{013}

Counter:

$K^{000} = K^{001} + J^{008}K^{002}$

B04 (22) K^{001} K^{003}

$K^{001} = K^{000} + J^{008}K^{003}$

B04 (22) K^{000} K^{002} A^{001} A^{003}

$K^{002} = K^{003} + J^{010}K^{001}$

A05 (22) K^{003} K^{000}

$K^{003} = K^{002} + J^{010}K^{000}$

A05 (22) K^{002} K^{001} K^{010} K^{011}

$K^{010} = K^{011} + J^{008}K^{003}K^{012}$

B05 (22) K^{011} K^{013}

$K^{011} = K^{010} + J^{008}K^{003}$

B05 (22) K^{010} K^{012} A^{002} A^{003}

$K^{012} = K^{013} + J^{010}K^{011}$

B06 (22) K^{013} K^{010}

$K^{013} = K^{012} + J^{010}K^{010}$

B06 (22) K^{012} K^{011}

Binary Adder:

$A^{001} = K^{001}$

A06 (21) A^{003} A^{004}

$A^{002} = K^{011}$

A06 (21) A^{003} A^{004}

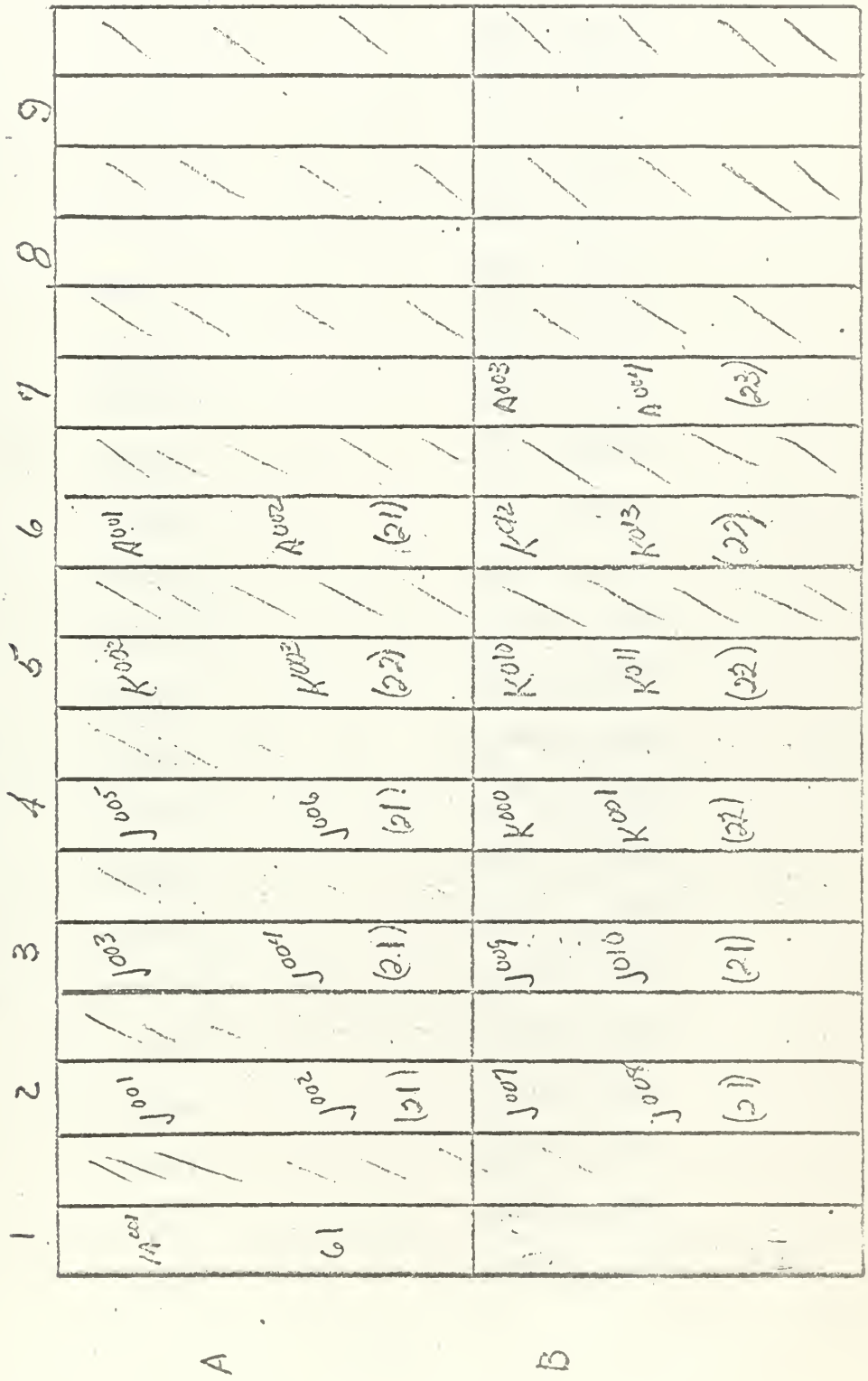
$A^{003} = K^{001}K^{011} + A^{001}A^{002}$

B07 (22) Sum output at test point A

$A^{004} = A^{001} + A^{002}$

B07 (22) Carry output at test point C

6.3 Chassis Map



6.4 Wire Tabulation

Origin	Destination	Description
A02-1	Signal Gen.	Input to J^{001}
A02-7	A02-6	$J^{002} - J^{001}$
A03-1	A02-12	$J^{003} - J^{002}$
A03-7	A03-6	$J^{004} - J^{003}$
A04-1	A03-12	$J^{005} - J^{004}$
B02-1	A02-5	$J^{007} - J^{001}$
	A03-11	$J^{007} - J^{001}$
B02-7	B02-6	$J^{008} - J^{007}$
B03-1	A02-11	$J^{009} - J^{002}$
	A04-5	$J^{009} - J^{005}$
B03-7	B03-6	$J^{010} - J^{009}$
B04-1	B04-12	$K^{000} - K^{001}$
B04-2	B02-12	$K^{000} - J^{008}$
	A05-5	$K^{000} - K^{002}$
B04-7	B04-6	$K^{001} - K^{000}$
B04-8	B02-11	$K^{001} - J^{008}$
	A05-11	$K^{001} - K^{003}$
A05-1	A05-12	$K^{002} - K^{003}$
A05-2	B03-12	$K^{002} - J^{010}$
	B04-11	$K^{002} - K^{001}$
A05-7	A05-6	$K^{003} - K^{002}$
A05-8	B03-11	$K^{003} - J^{010}$
	B04-5	$K^{003} - K^{000}$
B05-1	B05-12	$K^{010} - K^{011}$

B05-2	B02-10	$K^{010} - J^{008}$
	A05-10	$K^{010} - K^{003}$
	B06-5	$K^{010} - K^{012}$
B05-7	B05-6	$K^{011} - K^{010}$
B05-8	B02-9	$K^{011} - J^{008}$
	A05-9	$K^{011} - K^{003}$
B06-1	B06-12	$K^{012} - K^{013}$
B06-2	B03-10	$K^{012} - J^{010}$
	B05-11	$K^{012} - K^{011}$
B06-7	B06-6	$K^{013} - K^{012}$
B06-8	B03-9	$K^{013} - J^{010}$
	B05-5	$K^{013} - K^{010}$
A06-1	B04-10	$A^{001} - K^{001}$
A06-7	B05-10	$A^{002} - K^{011}$
B07-1	B04-9	$A^{003} - K^{001}$
	B05-9	$A^{003} - K^{011}$
B07-2	A06-6	$A^{003} - A^{001}$
	A06-12	$A^{003} - A^{002}$
B07-7	A06-5	$A^{004} - A^{001}$
B07-8	A06-11	$A^{004} - A^{002}$

7. Conclusion

Formal logical design techniques can become important "tools" for the logical designer if they can be re ted to the basic circuit module of the equipment being designed. Only two such techniques have been considered in this paper; the symbolic logic diagram, and Boolean algebra. Many other formal design techniques need to receive the same or similar treatment. For example, Boolean Matrices and Designation Numbers have received intense formal development and yet are seldom used in practical design problems.

BIBLIOGRAPHY

1. R. S. Ledley, Mathematical Foundations and Computational Methods for a Digital Logic Machine, J. Operations Research Soc. Am., vol. 2, no. 3, pp. 249-279, Aug. 1954.

2. E. J. McCluskey, Jr., Minimization of Boolean Functions, Bell System Technical Journal, vol. 35, pp. 1-28, Nov. 1956.

3. J. P. Roth, Algebraic Topological Methods for the Synthesis of Switching System I, Trans. Am. Math. Soc., vol. 88, no. 2, pp. 301-326, July 1958.

4. R. A. Higonnet and R. A. Grea, Logical Design of Electrical Circuits, McGraw Hill Book Company, Inc., New York, 1958.

5. G. H. Mealy, A Method for Synthesizing Sequential Circuits, Bell System Technical Journal, vol. 34, pp. 1045-1080, September 1955.

6. S. H. Caldwell, Switching Circuits and Logical Design, John Wiley and Sons, Inc., New York, 1958.

7. R. S. Ledley, Digital Computer and Control Engineering, McGraw Hill Book Company, Inc., New York, 1960.

8. E. M. McCormick, Digital Computer Primer, McGraw Hill Book Company, Inc., New York, 1959.

9. M. Phister, Jr., Logical Design of Digital Computers, John Wiley and Sons, Inc., New York, 1958.

10. J. R. Farrell, Logical Design of Digital Interface, USNPS Engineering Report 13-61, June 1961.

11. 1604 Computer Instruction Book, Vol. 1, Description and Operation, Publication 031, Control Data Corp., Minneapolis, Minn.

12. 1604 Computer Instruction Book, Vol. 2, Principles of Operation, Publication 032, Control Data Corp., Minneapolis, Minn.

13. 1604 Computer Input-Output Specification, Publication 030, Control Data Corp., Minneapolis, Minn.

14. 160 Computer Input-Output Specification, Publication 025, Control Data Corp., Minneapolis, Minn.

15. 160 Computer Programming Manual, Publication 023a, Control Data Corp., Minneapolis, Minn.

16. 160 Computer Instruction Book, Vol. 1, Description and Operation, Publication 068, Control Data Corp., Minneapolis, Minn.

APPENDIX A

Equations of Boolean Algebra

1.	$A + A = A$	Absorption rule for +
2.	$A \cdot A = A$	Absorption rule for ·
3.	$A + B = B + A$	Commutative rule for +
4.	$A \cdot B = B \cdot A$	Commutative rule for ·
5.	$(A + B) + C = A + (B + C) = A + B + C$	Associative rule for +
6.	$(A \cdot B) \cdot C = A \cdot (B \cdot C) = A \cdot B \cdot C$	Associative rule for ·
7.	$A \cdot (B + C) = A \cdot B + A \cdot C$	Distributive rule of · over +
8.	$A + B \cdot C = (A + B) \cdot (A + C)$	Distributive rule of + over ·
9.	$\overline{A \cdot B} = \overline{A} + \overline{B}$	De Morgan's rule for ·
10.	$\overline{A + B} = \overline{A} \cdot \overline{B}$	De Morgan's rule for +
11.	$A + I = I$	
12.	$A \cdot I = A$	
13.	$0 + A = A$	
14.	$0 \cdot A = A$	
15.	$\overline{A} + A = I$	
16.	$A \cdot \overline{A} = 0$	
17.	$A + A \cdot B = A$	
18.	$A + \overline{A} \cdot B = A + B$	
19.	$A \cdot B + B \cdot C + C \cdot A = (A + B) \cdot (B + C) \cdot (C + A)$	
20.	$A \cdot \overline{B} + A \cdot C + B \cdot C = A \cdot \overline{B} + B \cdot C$	
21.	$\overline{\overline{A}} = A$	

thesF2294
Engineering design of logical circuits.
3 2768 002 13383 7
DUDLEY KNOX LIBRARY